PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-158972

(43)Date of publication of application : 31.05.2002

(51)Int.Cl. H04N 5/92

H04N 5/85

H04N 5/91

H04N 5/93

(21)Application number : 2001-091830

(22)Date of filing : 28.03.2001

(71)Applicant : SONY CORP

(72)Inventor : KATO MOTOKI

HAMADA TOSHIYA

(30)Priority

Priority number :	2000183771	Priority date :	21.04.2000	Priority country :	JP
	2000271552		07.09.2000		JP

(54) INFORMATION PROCESSOR AND PROCESSING METHOD, AND RECORDING MEDIUM THEREFOR, AND PROGRAM AND ITS RECORDING MEDIUM

(57)Abstract:

PROBLEM TO BE SOLVED: To enable to manage commonly AV stream data recorded by analyz ing the position of an I picture and AV stream data recorded without analyzing the position of the I picture.

SOLUTION: CPI-type is described in PlayList(). The CPI type includes EP map type and TU map type. The EP map is used when the position of an I picture can be analyzed, and the TU map is used when the position of the I picture cannot be analyzed.

LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

CLAIMS

[Claim(s)]

[Claim 1] In the information processor which records AV stream data on a record medium The 1st table which describes the response relation between a presentation time stump and the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport

Paquette's arrival time, The 1st generation means which generates the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it, The information processor characterized by having a selection means to choose either said 1st table or said 2nd table according to the record approach, and the 1st record means which records said selected table on said record medium with said AV stream data.

[Claim 2] It is the information processor according to claim 1 which said 1st table is EP_map and is characterized by said 2nd table being TU_map.

[Claim 3] Said selection means is an information processor according to claim 1 characterized by choosing said 2nd table in the case of non cog NIZANTO record.

[Claim 4] Said selection means is an information processor given in a publication at claim 1 characterized by choosing said 1st table in the case of self encoding record.

[Claim 5] Said selection means is an information processor given in a publication at claim 1 characterized by choosing said 1st table in the case of cog NIZANTO record.

[Claim 6] The 2nd generation means which generates the playback assignment information that playback of said AV stream data is specified, It has further the 2nd record means which records said playback assignment information generated by said 2nd generation means on said record medium. Said playback assignment information An information processor given in a publication at claim 1 characterized by including the classification information which shows whether the hour entry of the playback section of said AV stream data is expressed in presentation time base, or it expresses in arrival time base.

[Claim 7] It is an information processor given in a publication at claim 6 characterized by for said playback assignment information to express the hour entry of the playback section of said AV stream data in presentation time base when said 1st table is recorded with said AV stream data, and for said playback assignment information to express the hour entry of the playback section of said AV stream data in arrival time base when said 2nd table is

recorded with said AV stream data.

[Claim 8] In the information processing approach of the information processor which records AV stream data on a record medium The 1st table which describes the response relation between a presentation time stump and the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, The generation step which generates the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it, The information processing approach characterized by including the selection step which chooses either said 1st table or said 2nd table according to the record approach, and the record step which records said selected table on said record medium with said AV stream data.

[Claim 9] In the program of the information processor which records AV stream data on a record medium The 1st table which describes the response relation between a presentation time stump and the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, The generation step which generates the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it, The selection step which chooses either said 1st table or said 2nd table according to the record approach, The record medium with which the program which the computer characterized by including the record step which records said selected table on said record medium with said AV stream data can read is recorded.

[Claim 10] To the computer which controls the information processor which records AV stream data on a record medium, a presentation time stump, The 1st table which describes response relation with the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, The generation step which generates the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it, The program which performs the selection step which chooses either said 1st table or said 2nd table according to the record approach, and the record step which

records said selected table on said record medium with said AV stream data.
[Claim 11] In the information processor which reproduces AV stream data from a record medium The 1st table which describes the response relation between a presentation time stump and the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, One side of the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it The information processor characterized by having a playback means to reproduce either said 1st table or said 2nd table, and the control means which controls the output of said AV stream data based on said reproduced table from said record medium currently recorded according to the record approach.
[Claim 12] In the information processing approach of the information processor which reproduces AV stream data from a record medium The 1st table which describes the response relation between a presentation time stump and the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, One side of the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it The playback step which reproduces either said 1st table or said 2nd table from said record medium currently recorded according to the record approach, The information processing approach characterized by including the control step which controls the output of said AV stream data based on said reproduced table.
[Claim 13] In the program of the information processor which reproduces AV stream data from a record medium The 1st table which describes the response relation between a presentation time stump and the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, One side of the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it The playback step which reproduces either said 1st table or said 2nd table from said record medium currently recorded according to the record approach, The record medium with which

the program which the computer characterized by including the control step which controls the output of said AV stream data based on said reproduced table can read is recorded.

[Claim 14] To the computer which controls the information processor which reproduces AV stream data from a record medium, a presentation time stump, The 1st table which describes response relation with the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, One side of the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it The program which performs the playback step which reproduces either said 1st table or said 2nd table, and the control step which controls the output of said AV stream data based on said reproduced table from said record medium currently recorded according to the record approach.

[Claim 15] In the record medium with which AV stream data are recorded A presentation time stump, The 1st table which describes response relation with the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, The record medium with which one side of the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it is characterized by what is recorded according to the record approach.

[Claim 16] The information processor carry out having a generation means generate the playback assignment information constituted in the information processor which records AV stream data on a record medium by the 1st information which shows the main playback pass, and the 2nd information which show the playback pass of ** reproduced synchronizing with said main playback pass, and said AV stream data and a record means record said playback assignment information on said record medium as the description.

[Claim 17] The playback pass of said ** is an information processor according to claim 16 characterized by being the pass for after recording of audio data.

[Claim 18] It is the information processor according to claim 16 which said 1st information is Main_path and is characterized by said 2nd information being

Sub_path.

[Claim 19] Said 2nd information is the information processor according to claim 16 characterized by to be included the time of day on said main pass which the Inn point of the file name of the type information showing the type of the playback pass of said ** and said AV stream which the playback pass of said ** refers to, and said AV stream of the playback pass of said **, an out point, and the Inn point of said playback pass synchronized and start on the time-axis of said main pass.

[Claim 20] The information-processing approach of carrying out containing the generation step which generates the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which show the playback pass of ** reproduced synchronizing with said main playback pass in the information-processing approach of the information processor which records AV stream data on a record medium, and said AV stream data and the record step which record to said record medium in said playback assignment information as the description.

[Claim 21] In the program of the information processor which records AV stream data on a record medium The generation step which generates the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which shows the playback pass of ** reproduced synchronizing with said main playback pass, The record medium with which the program which the computer characterized by including said AV stream data and the record step which records said playback assignment information on said record medium can read is recorded.

[Claim 22] The program which performs the generation step which generates the playback assignment information constituted by the 1st information which shows the main playback pass to the computer which controls the information processor which records AV stream data on a record medium, and the 2nd information which show the playback pass of ** reproduced synchronizing with said main playback pass, and said AV stream data and the record step which record to said record medium in said playback assignment information.

[Claim 23] The information processor carry out having the playback means

reproduce the playback assignment information constituted from a record medium in the information processor which reproduces AV stream data by the 1st information which shows the main playback pass, and the 2nd information which show the playback pass of ** reproduced synchronizing with said main playback pass from said record medium, and the control means control the output of said AV stream data based on said reproduced playback assignment information as the description.

[Claim 24] In the information processing approach of the information processor which reproduces AV stream data from a record medium The playback step which reproduces the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which shows the playback pass of ** reproduced synchronizing with said main playback pass from said record medium, The information processing approach characterized by including the control step which controls the output of said AV stream data based on said reproduced playback assignment information.

[Claim 25] In the program of the information processor which reproduces AV stream data from a record medium The playback step which reproduces the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which shows the playback pass of ** reproduced synchronizing with said main playback pass from said record medium, The record medium with which the program which the computer characterized by including the control step which controls the output of said AV stream data based on said reproduced playback assignment information can read is recorded.

[Claim 26] The program which performs the playback step which reproduces the playback assignment information constituted by the 1st information which shows the main playback pass to the computer which controls the information processor which reproduces AV stream data from a record medium, and the 2nd information which show the playback pass of ** reproduced synchronizing with said main playback pass from said record medium, and the control step control the output of said AV stream data based on said reproduced playback assignment information.

[Claim 27] The record medium characterized by recording the playback assignment information constituted in the record medium with which AV stream data are recorded by the 1st information which shows the main playback pass, and the 2nd information which shows the playback pass of ** reproduced synchronizing with said main playback pass.

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[Field of the Invention] Especially this invention relates to a record medium at the information processor which records a file including information, such as the information which indicates by explanation about a record medium in an information processor, the playback approach, a record medium, a program, and a list at GUI etc., the information on the main salvage pathways, the information on the salvage pathway of **, an initial entry during each playback section which constitutes the main salvage pathways, and the bookmark which sets to the scene for which the user asked, the information on a resume point the playback approach, a record medium, a program, and a list

[0002]

[Description of the Prior Art] In recent years, various kinds of optical disks are being proposed as a record medium of a dismountable disk mold from a record regenerative apparatus. The optical disk in which such record is possible is proposed as several G bytes of mass media, and its expectation as media which record AV (Audio Visual) signals, such as a video signal, is high. As the source (supply source) of digital AV signal recorded on the optical disk in which this record is possible, there are CS digital satellite broadcasting and BS digital broadcasting, and the terrestrial television broadcasting of a digital method etc. is proposed in the future.

[0003] Here, as for the digital video signal supplied from these sources, it is

common that picture compression is usually carried out by MPEG(Moving Picture Experts Group) 2 method. Moreover, the record rate of the equipment proper is set to the recording device. By the conventional noncommercial image are recording media, if it is an analog recording method when recording the digital video signal of the digital-broadcasting origin, it will record by carrying out a band limit after decoding a digital video signal. Or if it is digital storage methods including MPEG1 Video, MPEG 2 Video, and DV method, after being decoded once, by the record rate and coding method of the equipment proper, it will be re-encoded and will be recorded.

[0004] However, such a record approach decodes the supplied bit stream once, and it is accompanied by degradation of image quality in order to record by performing band limit and re-encoding after that. When the transmission rate of the digital signal inputted when the digital signal by which picture compression was carried out was recorded does not exceed the record rate of a record regenerative apparatus, degradation of image quality has few decodings and approaches of recording as it is in the supplied bit stream, without re-encoding. However, when the transmission rate of the digital signal by which picture compression was carried out exceeds the record rate of the disk as a record medium, it is necessary to record by carrying out re-encoding so that a transmission rate may become below the upper limit of the record rate of a disk after decoding with a record regenerative apparatus.

[0005] Moreover, when the bit rate of an input digital signal is transmitted by the adjustable rate method fluctuated by time amount, since a rotary head is a fixed engine speed, compared with the tape recording system with which a record rate turns into a fixed rate, data are once stored in a buffer, and the disk recording device which can do record burstily can use the capacity of a record medium without futility.

[0006] As mentioned above, in the future when digital broadcasting becomes in use, it is predicted that decoding and the record regenerative apparatus which recorded without re-encoding and used the disk as a record medium are asked for a broadcast signal like a data streamer with a digital signal.

[0007]

[Problem(s) to be Solved by the Invention] By the way, in order to be able to

perform high-speed playback when recording AV stream data on a record medium with a recording apparatus which was mentioned above for example, A stream data are analyzed, the location of I picture may be detected, and A stream data may not be analyzed as the case where it records on it as I picture can be accessed, but it may record as it is.

[0008] In such a case, the application program of dedication is prepared former, respectively, and it is alike, respectively and he was trying to record AV stream on a record medium more as an AV stream (AV stream in which high-speed playback is possible, or impossible AV stream) of a different format. Consequently, the technical problem which requires costs and time amount for development of an application program occurred. Moreover, since AV stream recorded by each application program was with the thing of a different format, the mutual compatibility of it was lost and it had the technical problem it becomes impossible to reproduce with common equipment.

[0009] Furthermore, in the conventional recording apparatus, the technical problem that the so-called thing [postrecording] was difficult occurred audio data, for example.

[0010] This invention is made in view of such a situation, and the 1st object is in enabling it to manage in common AV stream in which high-speed playback is possible, and impossible AV stream.

[0011] Furthermore, the 2nd object is to make after recording possible.

[0012]

[Means for Solving the Problem] The 1st information processor of this invention A presentation time stump, The 1st table which describes response relation with the address in AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, The 1st generation means which generates the 2nd table which describes response relation with the address in AV stream data of transport Paquette corresponding to it, It is characterized by having a selection means to choose either the 1st table or the 2nd table according to the record approach, and the 1st record means which records the selected table on a record medium with AV stream data.

[0013] Said 1st table is EP_map and the 2nd table can be made into TU_map.

[0014] Said selection means can choose the 2nd table in the case of non cog NIZANTO record.
[0015] Said selection means can choose the 1st table in the case of self encoding record.
[0016] Said selection means can choose the 1st table in the case of cog NIZANTO record.
[0017] It has further the 2nd record means which records the playback assignment information generated by the 2nd generation means which generates the playback assignment information that playback of said AV stream data is specified, and the 2nd generation means on a record medium, and the classification information shown [whether playback assignment information expresses the hour entry of the playback section of AV stream data in presentation time base or it expresses in arrival time base, and] can contain.
[0018] When the 1st table is recorded with said AV stream data, playback assignment information expresses the hour entry of the playback section of AV stream data in presentation time base, and when the 2nd table is recorded with AV stream data, playback assignment information can express the hour entry of the playback section of AV stream data in arrival time base.
[0019] The 1st information processing approach of this invention A presentation time stump, The 1st table which describes response relation with the address in AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, The generation step which generates the 2nd table which describes response relation with the address in AV stream data of transport Paquette corresponding to it, It is characterized by including the selection step which chooses either the 1st table or the 2nd table according to the record approach, and the record step which records the selected table on a record medium with AV stream data.
[0020] The program of the 1st record medium of this invention A presentation time stump, The 1st table which describes response relation with the address in AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, The generation step which generates the 2nd table which describes response relation with the address in

AV stream data of transport Paquette corresponding to it, It is characterized by including the selection step which chooses either the 1st table or the 2nd table according to the record approach, and the record step which records the selected table on a record medium with AV stream data.

[0021] The 1st program of this invention A presentation time stump, The 1st table which describes response relation with the address in AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, The generation step which generates the 2nd table which describes response relation with the address in AV stream data of transport Paquette corresponding to it, The selection step which chooses either the 1st table or the 2nd table according to the record approach, and the record step which records the selected table on a record medium with AV stream data are performed.

[0022] The 2nd information processor of this invention A presentation time stump, The 1st table which describes response relation with the address in AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, One side of the 2nd table which describes response relation with the address in AV stream data of transport Paquette corresponding to it It is characterized by having a playback means to reproduce either the 1st table or the 2nd table, and the control means which controls the output of AV stream data based on the reproduced table from the record medium currently recorded according to the record approach.

[0023] The 2nd information processing approach of this invention A presentation time stump, The 1st table which describes response relation with the address in AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, One side of the 2nd table which describes response relation with the address in AV stream data of transport Paquette corresponding to it It is characterized by including the playback step which reproduces either the 1st table or the 2nd table, and the control step which controls the output of AV stream data based on the reproduced table from the record medium currently recorded according to the record approach.

[0024] The program of the 2nd record medium of this invention A presentation time stump, The 1st table which describes response relation with the address in AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, One side of the 2nd table which describes response relation with the address in AV stream data of transport Paquette corresponding to it It is characterized by including the playback step which reproduces either the 1st table or the 2nd table, and the control step which controls the output of AV stream data based on the reproduced table from the record medium currently recorded according to the record approach.

[0025] The 2nd program of this invention A presentation time stump, The 1st table which describes response relation with the address in AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, One side of the 2nd table which describes response relation with the address in AV stream data of transport Paquette corresponding to it The program which performs the playback step which reproduces either the 1st table or the 2nd table, and the control step which controls the output of AV stream data based on the reproduced table from the record medium currently recorded according to the record approach.

[0026] The 1st record medium of this invention is characterized by recording one side of the 2nd table which describes the response relation between the 1st table which describes the response relation between a presentation time stump and the address in AV stream data of the access unit corresponding to it or the arrival time stump based on transport Paquette's arrival time, and the address in AV stream data of transport Paquette corresponding to it according to the record approach.

[0027] The 3rd information processor of this invention carries out having a generation means generate the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which shows the playback pass of ** reproduced synchronizing with the main playback pass, and AV stream data and a record means record playback assignment information on a record medium as the description.

[0028] The playback pass of said ** can be considered as the pass for after

recording of audio data.

[0029] Said 1st information is Main_path and the 2nd information can be made into Sub_path.

[0030] Said 2nd information can contain the time of day on the main pass which the Inn point of the file name of the type information showing the type of the playback pass of ** and AV stream which the playback pass of ** refers to, and AV stream of the playback pass of **, an out point, and the Inn point of playback pass synchronize and start on the time-axis of the main pass.

[0031] The 3rd information-processing approach of this invention carries out containing the generation step which generates the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which shows the playback pass of ** reproduced synchronizing with the main playback pass, and AV stream data and the record step which record playback assignment information on a record medium as the description.

[0032] The program of the 3rd record medium of this invention carries out containing the generation step which generates the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which shows the playback pass of ** reproduced synchronizing with the main playback pass, and AV stream data and the record step which record playback assignment information on a record medium as the description.

[0033] The 3rd program of this invention performs the generation step which generates the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which shows the playback pass of ** reproduced synchronizing with the main playback pass, and AV stream data and the record step which record to a record medium in playback assignment information.

[0034] Said 4th information processor carries out having a playback means reproduce the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which show the playback pass of ** reproduced synchronizing with said main playback pass from said record medium, and the control means which control

in the output of said AV stream data based on said reproduced playback assignment information as the description.

[0035] The 4th information-processing approach of this invention carries out containing the playback step which reproduces the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which show the playback pass of ** reproduced synchronizing with the main playback pass from a record medium, and the control step control the output of AV stream data based on the reproduced playback assignment information as the description.

[0036] The program of the 4th record medium of this invention carries out containing the playback step which reproduces the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which show the playback pass of ** reproduced synchronizing with the main playback pass from a record medium, and the control step control the output of AV stream data based on the reproduced playback assignment information as the description.

[0037] The 4th program of this invention performs the playback step which reproduces the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which show the playback pass of ** reproduced synchronizing with the main playback pass from a record medium, and the control step control the output of AV stream data based on the reproduced playback assignment information.

[0038] The 2nd record medium of this invention is characterized by recording the playback assignment information constituted by the 1st information which shows the main playback pass, and the 2nd information which shows the playback pass of ** reproduced synchronizing with the main playback pass.

[0039] It sets to a record medium at the 1st information processor of this invention and an approach, the program of a record medium, a program, and a list. The 1st table which describes the response relation between a presentation time stump and the address in said AV stream data of the access unit corresponding to it, Or one side of the 2nd table which describes the response relation between the arrival time stump based on transport Paquette's arrival time and the address in said AV stream data of transport

Paquette corresponding to it is recorded according to the record approach.
[0040] It sets to a program at the 2nd information processor of this invention and an approach, the program of a record medium, and a list. The 1st table which describes the response relation between a presentation time stump and the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, From the record medium with which one side of the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it is recorded according to the record approach, the table is reproduced and an output is controlled based on it.
[0041] The playback assignment information constituted by the 1st information which shows the main playback pass in the 2nd record medium, and the 2nd information which shows the playback pass of ** reproduced synchronizing with said main playback pass is recorded on the 3rd information processor of this invention and an approach, the program of a record medium, a program, and a list.
[0042] The 1st information which shows the main playback pass in a program, and the playback assignment information constituted using the 2nd information which shows the playback pass of ** reproduced synchronizing with said main playback pass are reproduced by the 4th information processor of this invention and an approach, the program of a record medium, and the list from a record medium, and an output is controlled based on it.
[0043]
[Embodiment of the Invention] Below, the gestalt of operation of this invention is explained with reference to a drawing. Drawing 1 is drawing showing the example of an internal configuration of the record regenerative apparatus 1 which applied this invention. First, the configuration of the part which performs actuation which records the signal inputted from the outside on a record medium is explained. The record regenerative apparatus 1 is considered as the configuration which can input analog data or digital data and can be recorded.
[0044] The video signal of an analog is inputted into a terminal 11, and the audio signal of an analog is inputted into a terminal 12, respectively. The

video signal inputted into the terminal 11 is outputted to the analysis section 14 and the AV encoder 15, respectively. The audio signal inputted into the terminal 12 is outputted to the AV encoder 15. The analysis section 14 extracts the focus, such as a scene change, from the inputted video signal.

[0045] The AV encoder 15 encodes the video signal and audio signal which were inputted, respectively, and outputs system information (S), such as a coding video stream (V), a coding audio stream (A), and AV synchronization, to a multiplexer 16.

[0046] A coding video stream is a video stream encoded for example, by MPEG(Moving Picture Expert Group) 2 method, and coding audio streams are the audio stream encoded for example, by MPEG1 method, an audio stream encoded by DORUBI AC3 method. A multiplexer 16 multiplexes the stream of the inputted video and an audio based on input system information, and outputs it to the multiplexing stream analysis section 18 and sow spa KETTAIZA 19 through a switch 17.

[0047] Multiplexing streams are for example, an MPEG 2 transport stream and an MPEG 2 program stream. Sow spa KETTAIZA 19 encodes AV stream which consists of source packets in the inputted multiplexing stream according to an application format of the record medium 100 on which the stream is made to record. Processing predetermined in the ECC (error correction) coding section 20 and the modulation section 21 is performed, and AV stream is outputted to the write-in section 22. The write-in section 22 writes AV stream file in a record medium 100 based on the control signal outputted from a control section 23 (it records).

[0048] Transport streams, such as digital television broadcast inputted from a digital interface or a digital television tuner, are inputted into a terminal 13. They are the method which records those with two kind, and them on transparent at the recording method of a transport stream inputted into the terminal 13, and the method recorded after carrying out re-encoding for the objects, such as lowering a record bit rate. The directions information on a recording method is inputted into a control section 23 from the terminal 24 as a user interface.

[0049] When recording an input transport stream on transparent, the transport

stream inputted into the terminal 13 is outputted to the multiplexing stream analysis section 18 and sow spa KETTAIZA 19. Since processing until AV stream is recorded on the record medium 100 after this is the same processing as the case where above-mentioned input audio osmosis and a video signal are encoded and recorded, the explanation is omitted.

[0050] When recording after re-encoding an input transport stream, the transport stream inputted into the terminal 13 is inputted into a demultiplexer 26. A demultiplexer 26 performs demultiplex processing to the inputted transport stream, and extracts a video stream (V), an audio stream (A), and system information (S).

[0051] A video stream is outputted to the AV decoder 27 among the streams (information) extracted by the demultiplexer 26, and an audio stream and system information are outputted to a multiplexer 16, respectively. The AV decoder 27 decodes the inputted video stream, and outputs the playback video signal to the AV encoder 15. The AV encoder 15 encodes an input video signal, and outputs a coding video stream (V) to a multiplexer 16.

[0052] On the other hand, based on input system information, the audio stream which was outputted from the demultiplexer 26 and inputted into the multiplexer 16, system information, and the video stream outputted from the AV encoder 15 are multiplexed, and is outputted to the multiplexing stream analysis section 18 and source packet TAIZA 19 through a switch 17 as a multiplexing stream. Since processing until AV stream is recorded on the record medium 100 after this is the same processing as the case where an above-mentioned input audio signal and an above-mentioned video signal are encoded and recorded, the explanation is omitted.

[0053] The record regenerative apparatus 1 of the gestalt of this operation also records the application database information that the file is explained while recording the file of AV stream on a record medium 100. Application database information is created by the control section 23. The input to a control section 23 is the description information on the dynamic image from the analysis section 14, the description information on AV stream from the multiplexing stream analysis section 18, and the directions information from a user that it is inputted from a terminal 24.

[0054] the description information on the dynamic image supply from the analysis section 14 be the information related to the characteristic image in an input dynamic image signal, it be assignment information (mark), such as a start point of a program, a point change [scene], and a point of a commercials (CM) end [initiation /], and the information on the thumbnail image of the image of the appointed location be also include.

[0055] The description information on AV stream from the multiplexing stream analysis section 18 is the information related to the encoded information of AV stream recorded, for example, is the changing point information on the address information of I picture in AV stream, the coding parameter of AV stream, and the coding parameter in AV stream, the information (mark) related to the characteristic image in a video stream, etc.

[0056] The directions information of the user from a terminal 24 is the character alphabetic character explaining the assignment information on the playback section specified by the user in AV stream, and the content of the playback section, the bookmark which a user sets to a favorite scene, the information on a resume point, etc.

[0057] A control section 23 creates the management information (info.dvr) of the database (Clip) of AV stream, the database of what (PlayList) carried out grouping of the playback section (PlayItem) of AV stream, and the content of record of a record medium 100, and the information on a thumbnail image based on the above-mentioned input. Like AV stream, the application database information which consists of such information is processed in the ECC coding section 20 and the modulation section 21, and is inputted into the write-in section 22. The write-in section 22 records a database file on a record medium 100 based on the control signal outputted from a control section 23.

[0058] The detail about the application database information mentioned above is mentioned later.

[0059] Thus, when application database information is reproduced with AV stream file (file of image data and voice data) recorded on the record medium 100, a control section 23 directs to read application database information from a record medium 100 to the read-out section 28 first. And the read-out section 28 reads application database information from a record medium 100, and the

application database information is inputted into a control section 23 through processing of the recovery section 29 and the ECC decode section 30.

[0060] A control section 23 outputs the list of PlayList currently recorded on the record medium 100 to the user interface of a terminal 24 based on application database information. A user chooses PlayList to reproduce from the list of PlayList, and the information about PlayList which had playback specified is inputted into a control section 23. A control section 23 directs read-out of AV stream file required for playback of the PlayList in the read-out section 28. The read-out section 28 reads AV stream which corresponds from a record medium 100 according to the directions, and outputs it to the recovery section 29. It gets over by performing predetermined processing, and AV stream inputted into the recovery section 29 is further outputted source DEPAKETTAIZA 31 through processing of the ECC decode section 30.

[0061] Reading appearance of source DEPAKETTAIZA 31 is carried out from a record medium 100, and it is changed into the stream which can output AV stream of an application format to which predetermined processing was performed to a demultiplexer 26. A demultiplexer 26 outputs system information (S), such as a video stream (V) which constitutes the playback section (PlayItem) of AV stream specified by the control section 23, an audio stream (A), and AV synchronization, to the AV decoder 27. The AV decoder 27 decodes a video stream and an audio stream, and outputs a playback video signal and a playback audio signal from the terminal 32 and terminal 33 which correspond, respectively.

[0062] Moreover, when the information which directs random access playback and special playback is inputted from the terminal 24 as a user interface, based on the content of the database (Clip) of AV stream, a control section 23 determines the read-out location of AV stream from a storage 100, and directs read-out of the AV stream in the read-out section 28. For example, when reproducing PlayList chosen by the user from predetermined time of day, as a control section 23 reads the data with the time stump nearest to the specified time of day from I picture, it is read, and it is directed in the section 28.

[0063] Moreover, when high-speed playback (Fast-forward playback) is directed by the user, as sequential continuation is carried out and a control

section 23 reads I-picture data in AV stream based on the database (Clip) of AV stream, it is read, and it is directed in the section 28.

[0064] The read-out section 28 reads the data of AV stream from the specified random access point, and the data by which reading appearance was carried out are reproduced through processing of latter each part.

[0065] Next, a user explains the case where AV stream currently recorded on the record medium 100 is edited. When a user wants to specify the playback section of AV stream currently recorded on the record medium 100, and to create new salvage pathway, For example, from the song program of Program A, reproduce Singer's A part and it continues after that. The information on the start point (Inn point) of the playback section and an ending point (out point) is inputted into a control section 23 from the terminal 24 as a user interface to create the salvage pathway of wanting to reproduce the part of the singer A of the song program of Program B. A control section 23 creates the database of what (PlayList) carried out grouping of the playback section (PlayItem) of AV stream.

[0066] When a user wants to eliminate a part of AV stream currently recorded on the record medium 100, the information on the Inn point of the elimination section and an out point is inputted into a control section 23 from the terminal 24 as a user interface. A control section 23 changes the database of PlayList so that only required AV stream part may be referred to. Moreover, it directs in the write-in section 22 so that the unnecessary stream part of AV stream may be eliminated.

[0067] It is the case where a user wants to specify the playback section of AV stream currently recorded on the record medium 100, and to create new salvage pathway, and the case where he wants to connect each playback section seamlessly is explained. In such a case, a control section 23 creates the database of what (PlayList) carried out grouping of the playback section (PlayItem) of AV stream, and performs near a node partial re-encoding and re-multiplexing of the playback section of a video stream further.

[0068] First, the information on the picture of the Inn point of the playback section and the information on the picture of an out point are inputted into a control section 23 from a terminal 24. A control section 23 directs read-out of

data required in order to reproduce the Inn point side picture and the picture by the side of an out point in the read-out section 28. And the read-out section 28 reads data from a record medium 100, and the data is outputted to a demultiplexer 26 through the recovery section 29, the ECC decode section 30, and source DEPAKETTAIZA 31.

[0069] A control section 23 analyzes the data inputted into the demultiplexer 26, determines a re-multiplex system as the re-encoding approach (modification of picture_coding_type, assignment of the re-encoded amount of coding bits) of a video stream, and supplies the method to the AV encoder 15 and a multiplexer 16.

[0070] Next, a demultiplexer 26 divides the inputted stream into a video stream (V), an audio stream (A), and system information (S). A video stream has "the data inputted into the AV decoder 27", and "the data inputted into a multiplexer 16." It is data required in order to re-encode the former data, and this is decoded by the AV decoder 27, and the decoded picture is re-encoded with the AV encoder 15, and is made into a video stream. The latter data are data copied from the stream of an original copy without carrying out re-encoding. About an audio stream and system information, it is directly inputted into a multiplexer 16.

[0071] Based on the information inputted from the control section 23, a multiplexer 16 multiplexes an input stream and outputs a multiplexing stream. A multiplexing stream is processed in the ECC coding section 20 and the modulation section 21, and is inputted into the write-in section 22. The write-in section 22 records AV stream on a record medium 100 based on the control signal supplied from a control section 23.

[0072] Explanation about actuation of the playback and edit based on application database information and its information is given to below. Drawing 2 is drawing explaining the structure of an application format. An application format has two layers, PlayList and Clip, for management of AV stream. Volume Information carries out management of all Clip(s) and PlayList(s) in a disk. Here, the pair of one AV stream and its attached information is considered to be one object, and it is called Clip. AV stream file calls Clip AV stream file, and the attached information is called Clip

Information file.

[0073] One Clip AV stream file stores the data which have arranged the MPEG 2 transport stream in the structure in which it is specified by application format. Generally, although a file is treated as a sequence of bytes, the contents of Clip AV stream file are developed on a time-axis, and the entry point in Clip is mainly specified in a hourly base. When the time stump of the access point to predetermined Clip is given, Clip Information file is useful in order to find the address information which should start read-out of data in Clip AV stream file.

[0074] PlayList is explained with reference to drawing 3 . PlayList chooses from Clip(s) the playback section which a user wants to see, and it is prepared in order to enable it to edit it simply. One PlayList is the meeting of the playback section in Clip. The one playback section in predetermined Clip is called PlayItem, and it is expressed with the pair of the Inn point on a time-axis (IN), and an out point (OUT). Therefore, PlayList is constituted when two or more PlayItem(s) gather.

[0075] There are two types of PlayList(s). One is Real PlayList and another is Virtual PlayList. Real PlayList is sharing the stream part of Clip which it is referring to. That is, when Real PlayList occupies in a disk the data volume equivalent to the stream part of Clip which is referring to it and Real PlayList is eliminated, data are eliminated also for the stream part of Clip which it is referring to.

[0076] Virtual PlayList is not sharing the data of Clip. Therefore, even if Virtual PlayList is changed or eliminated, by the content of Clip, change does not arise at all.

[0077] Next, edit of Real PlayList is explained. Drawing 4 (A) is drawing about the creation (create: creation) of Real PlayList, and when AV stream is recorded as new Clip, it is actuation in which Real PlayList which refers to the whole Clip is newly created.

[0078] Drawing 4 (B) is drawing about the divide (divide: division) of Real PlayList, and is actuation in which Real PlayList is divided at a point [****] and divided into two Real PlayList. when two programs are managed in one clip managed by one PlayList, a user wants to do the actuation of this division

again registration (record) as each program -- like -- it is sometimes carried out. There is nothing for which the content of Clip is changed by this actuation (the Clip itself is divided).

[0079] Drawing 4 (C) is drawing about the combined harvester and thresher (combine: association) of Real PlayList, and is actuation which combines two Real PlayList and is set to one new Real PlayList. a user wants, as for the actuation of this association, to reregister two programs as one program -- like -- it is sometimes carried out. There is nothing for which Clip is changed by this actuation (the Clip itself is set to one).

[0080] Drawing 5 (A) is drawing about deletion (delete: deletion) of whole Real PlayList, and when actuation which eliminates whole predetermined Real PlayList is carried out, the stream part to which Clip which deleted Real PlayList refers to corresponds is also deleted.

[0081] Drawing 5 (B) is drawing about partial deletion of Real PlayList, and when a part [**** / Real PlayList] is deleted, it is changed so that corresponding PlayItem may refer to only the stream part of required Clip. And the stream part to which Clip corresponds is deleted.

[0082] Drawing 5 (C) is drawing about minimization (Minimize: minimization) of Real PlayList, and is actuation of referring to only the stream part of Clip required for Virtual PlayList for PlayItem corresponding to Real PlayList. Virtual PlayList The stream part to which it takes and unnecessary Clip corresponds is deleted.

[0083] Real PlayList is changed by actuation which was mentioned above, when the stream part of Clip which the Real PlayList refers to is deleted, Virtual PlayList which is using the deleted Clip may exist, and a problem may arise by deleted Clip in the Virtual PlayList.

[0084] As opposed to actuation of [so that such a thing may not arise] deletion to a user "If Virtual PlayList which is referring to the stream part of Clip which the Real PlayList is referring to exists and the Real PlayList is eliminated although the Virtual PlayList will also be eliminated, is still it good? processing of the deletion with directions of a user after urging a check (warning) by displaying the message " etc. -- activation -- or it cancels. Or actuation of minimization is made to be performed instead of deleting Virtual

PlayList to Real PlayList.

[0085] Next, the actuation to Virtual PlayList is explained. The content of Clip is not changed even if actuation is performed to Virtual PlayList. Drawing 6 is an assemble (Assemble). Edit (IN-OUT edit) It is related drawing and is actuation of making PlayItem of the playback section for which it asked when the user wanted to see, and creating Virtual PlayList. The seamless connection between PlayItem(s) is supported by the application format (after-mentioned).

[0086] As shown in drawing 6 (A), two Real PlayList 1 and 2, When Clip 1 and 2 corresponding to each RealPlayList exists A user directs the predetermined section in Real PlayList1 (section-layItem1 to In1 thru/or Out1) as the playback section, and as the section reproduced continuously When the predetermined section in Real PlayList2 (section-layItem2 to In2 thru/or Out2) is directed as the playback section, As shown in drawing 6 (B), one Virtual PlayList which consists of PlayItem1 and PlayItem2 is created.

[0087] Next, Virtual PlayList A reorganization collection (Re-editing) is explained. A reorganization collection has insertion (insert) of modification of the Inn point in Virtual PlayList, and an out point, and new PlayItem to Virtual PlayList, an addition (append), deletion of PlayItem in Virtual PlayList, etc. Moreover, Virtual PlayList itself can also be deleted.

[0088] Drawing 7 is drawing about postrecording (Audio dubbing (post recording)) of the audio to Virtual PlayList, and is actuation which registers postrecording of the audio to Virtual PlayList as subpass. Postrecording of this audio is supported by the application format. An additional audio stream is added to AV stream of the main path of Virtual PlayList as subpass.

[0089] As actuation common to Real PlayList and Virtual PlayList, there is modification (Moving) of the playback sequence of PlayList as shown in drawing 8 . This actuation is modification of the playback sequence of PlayList in the inside of a disk (volume), and is supported by Table Of PlayList (with reference to drawing 20 etc., it mentions later) defined in an application format. As [change / by this actuation / the content of Clip]

[0090] Next, a mark (Mark) is explained. The mark is prepared in order to specify the highlights in Clip and PlayList, and characteristic time amount.

Specify the characteristic scene resulting from the content of the AV stream, for example, the mark added to Clip is a point changing [scene] etc. When reproducing PlayList, it can be used with reference to the mark of Clip which the PlayList refers to.

[0091] Are mainly set by the user, for example, the marks added to PlayList are a bookmark, a resume point, etc. Setting a mark to Clip or PlayList is performed by adding the time stump in which the time of day of a mark is shown to a mark list. Moreover, deleting a mark is removing the time stump of the mark out of a mark list. Therefore, as for AV stream, a change of what is not made by setting out or deletion of a mark, either.

[0092] Next, a thumbnail is explained. A thumbnail is a still picture added to Volume, PlayList, and Clip. There are two classes of thumbnails and one is a thumbnail as representation drawing showing the content. This is used in the menu screen for choosing the thing a user mainly wants to operate and look at cursor (un-illustrating) etc. Another is an image showing the scene which the mark has pointed out.

[0093] Volume and each Playlist need to enable it to have representation drawing. The representation drawing of Volume assumes being used when displaying the still picture showing the content of the disk first, when a disk (a record medium 100 presupposes that it is a disk-like thing, and is suitably described to be a disk a record medium 100 and the following) is set to the predetermined location of the record regenerative apparatus 1. The representation drawing of Playlist assumes being used as a still picture for expressing the content of Playlist in the menu screen which chooses Playlist.

[0094] Although it is possible as representation drawing of Playlist to make the image of the beginning of Playlist into a thumbnail (representation drawing), it is not not necessarily the image optimal when the image of the head of the playback time of day 0 expresses the content. Then, a user enables it to set up the image of arbitration as a thumbnail of Playlist. Two kinds of thumbnails are called a menu thumbnail above. Since a menu thumbnail is displayed frequently, reading appearance of it needs to be carried out to a high speed from a disk. For this reason, it is efficient to store all menu thumbnails in one file. It is not necessary to be necessarily the picture extracted from the

animation in volume, and as shown in drawing 10 , a menu thumbnail may be taken from a personal computer or a digital still camera, and a ********** image is sufficient as it.

[0095] It can be necessary to strike two or more marks, and in order to know the content of the mark location, it is necessary to enable it to see the image of a marking point easily to Clip and Playlist on the other hand. The picture showing such a marking point is called a mark thumbnail (Mark Thumbnails). Therefore, the image which becomes the origin of a thumbnail becomes main [what extracted the image of a marking point] from the image captured from the outside.

[0096] Drawing 11 is the mark attached to PlayList, and drawing showing the relation of the mark thumbnail, and drawing 12 is the mark attached to Clip, and drawing showing the relation of the mark thumbnail. Since a mark thumbnail is used by the sub menu etc. when the detail of Playlist is expressed unlike a menu thumbnail, what reading appearance is carried out in the short access time is not required. Therefore, it does not become a problem, even if it takes time amount somewhat because the record regenerative apparatus 1 reads an aperture and a part of its file for a file whenever a thumbnail is needed.

[0097] Moreover, in order to reduce the number of files which exists in volume, all mark thumbnails are good to store in one file. Although Playlist can have one menu thumbnail and two or more mark thumbnails, since Clip does not have the need that a direct user chooses (it usually specifies via Playlist), it does not need to prepare a menu thumbnail.

[0098] Drawing 13 is drawing having shown the relation of the menu thumbnail at the time of taking having mentioned above into consideration, a mark thumbnail, PlayList, and Clip. The menu thumbnail prepared in the menu thumbnail file for every PlayList is filed. The volume thumbnail representing the content of the data currently recorded on the disk is contained in the menu thumbnail file. The thumbnail by which the mark thumbnail file was created for every Clip with every PlayList is filed.

[0099] Next, CPI (Characteristic Point Information) is explained. CPI is data contained in a Clip information file, and when the time stump of the access

point to Clip is given, it is mainly used in order to find the data address which should start read-out of data in Clip AV stream file. Two kinds of CPI(s) are used with the gestalt of this operation. One is EP_map and another is TU_map.

[0100] EP_map is the list of entry point (EP) data, and it is extracted from an elementary stream and a transport stream. This has the address information for finding the location of the entry point which should start decoding in AV stream. One EP data consists of pairs of the data address in a presentation time stump (PTS) and AV stream of the access unit corresponding to the PTS.

[0101] EP_map is mainly used for two objects. It is used in order to find the data address in AV stream of the access unit referred to [1st] with a presentation time stump in PlayList. It is used for the 2nd for first forward playback or first reverse playback. When the record regenerative apparatus 1 records an input AV stream and the syntax of the stream can be analyzed, EP_map is created and it is recorded on a disk.

[0102] TU_map has the list of the time unit (TU) data based on transport Paquette's arrival time inputted through a digital interface. This gives the relation between the time amount of the arrival time base, and the data address in AV stream. When the record regenerative apparatus 1 records an input AV stream and the syntax of the stream cannot be analyzed, TU_map is created and it is recorded on a disk.

[0103] STCInfo stores the break point information on STC in AV stream file which is storing the MPEG 2 transport stream. When AV stream has the break point of STC, PTS of the same value may appear in the AV stream file. Therefore, when pointing out a certain time of day on AV stream with the PTS base, just PTS of an access point is inadequate in order to specify the point. Furthermore, the index of the STC section [****] containing the PTS is required. About the STC section [****], it is by this format. STC-sequence, a call, and its index are called STC-sequence-id. The information on STC-sequence is defined by STCInfo of Clip Information file. STC-sequence-id is an option in AV stream file which uses it by AV stream file with EP_map, and has TU_map.

[0104] A program is the meeting of an elementary stream and shares only one

system time base for synchronous playback of these streams. It is useful that the content of the AV stream is understood in advance of decoding of AV stream for a regenerative apparatus (record regenerative apparatus 1 of drawing 1). For example, they are information, such as a value of PID of transport Paquette who transmits the elementary stream of video or an audio, video, and a component class of audio, (for example, audio streams of the video of HDTV, and MPEG-2AAC etc.). This information is useful although the menu screen which explains to a user the content of PlayList which refers to AV stream is created, and it is useful in order to set AV decoder of a regenerative apparatus, and the initial state of a demultiplexer in advance of decoding of AV stream. For this reason, Clip Information file has ProgramInfo for explaining the content of the program.

[0105] As for AV stream file which is storing the MPEG 2 transport stream, the content of a program may change in a file. For example, it is that PID of transport Paquette who transmits a video elementary stream changes, or the component class of video stream changes from SDTV to HDTV etc.

[0106] ProgramInfo stores the information on the changing point of the content of a program in the inside of AV stream file. In AV stream file, the content of a program defined in this format calls the fixed section Program-sequence. Program-sequence is an option in AV stream file which uses it by AV stream file with EP_map, and has TU_map.

[0107] The gestalt of this operation defines the stream format (SESF) of self encoding. SESF is used when encoding to an MPEG 2 transport stream, after decoding the object which encodes an analog input signal, and a digital input signal (for example, DV).

[0108] SESF defines a coding limit of the elementary stream about an MPEG-2 transport stream and AV stream. When the record regenerative apparatus 1 encodes and records a SESF stream, EP_map is created and it is recorded on a disk.

[0109] Either of the methods shown below is used and the stream of digital broadcasting is recorded on a record medium 100. First, transformer coding of the stream of digital broadcasting is carried out at a SESF stream. In this case, the recorded stream must be based on SESF. In this case, EP_map must be

created and it must be recorded on a disk.

[0110] Or transformer coding is carried out at a new elementary stream, and the elementary stream which constitutes a digital-broadcasting stream is re-multiplexed to the new transport stream based on the stream format which the normalization organization of the digital-broadcasting stream defines. In this case, EP_map must be created and it must be recorded on a disk.

[0111] For example, an input stream is an MPEG-2 transport stream of ISDB (specification name of digital BS broadcast of Japan) conformity, and suppose that it contains a HDTV video stream and a MPEG AAC audio stream. Transformer coding of the HDTV video stream is carried out at a SDTV video stream, and the SDTV video stream and AAC audio stream of an original copy are re-multiplexed to TS. Both the transport streams recorded as a SDTV stream must be based on an ISDB format.

[0112] The stream of digital broadcasting is the case (it records without changing any input transport streams) where an input transport stream is recorded on transparent as other methods at the time of being recorded on a record medium 100, and EP_map is then created and it is recorded on a disk.

[0113] Or it is the case (it records without changing any input transport streams) where an input transport stream is recorded on transparent, and TU_map is then created and it is recorded on a disk.

[0114] Next, a directory and a file are explained. Hereafter, the record regenerative apparatus 1 is suitably described to be DVR (Digital Video Recording). Drawing 14 is drawing showing an example of the directory structure on a disk. Directories required on the disk of DVR are a root directory including a "DVR" directory, a "PLAYLIST" directory, a "CLIPINF" directory, and a "DVR" directory including an "M2TS" directory and "DATA" directory, as shown in drawing 14 . Although directories other than these may be made to be created under a root directory, they presuppose that it is ignored in an application format of the gestalt of this operation.

[0115] All the files and directories that are specified by DVR application format in the bottom of a "DVR" directory are stored. A "DVR" directory includes four directories. On the bottom of a "PLAYLIST" directory, the database file of Real PlayList and Virtual PlayList is put. This directory exists, even if one does not

have PlayList.

[0116] The database of Clip is put on the bottom of a "CLIPINF" directory. This directory also exists, even if Clip does not have one. AV stream file is put on the bottom of an "M2TS" directory. This directory exists, even if one does not have AV stream file. As for the "DATA" directory, the file of data broadcasting, such as digital TV broadcast, is stored.

[0117] A "DVR" directory stores the file shown below. It is made under a "info.dvr" file and a DVR directory, and the overall information on an application layer is stored. Only one info.dvr must be in the bottom of a DVR directory. A file name presupposes that it is fixed to info.dvr. A "menu.thmb" file stores the information relevant to a menu thumbnail image. Zero or one menu thumbnail must be in the bottom of a DVR directory. A file name presupposes that it is fixed to memu.thmb. When one does not have a menu thumbnail image, this file does not need to exist.

[0118] A "mark.thmb" file stores the information relevant to a mark thumbnail image. Zero or one mark thumbnail must be in the bottom of a DVR directory. A file name presupposes that it is fixed to mark.thmb. When one does not have a menu thumbnail image, this file does not need to exist.

[0119] A "PLAYLIST" directory stores two kinds of PlayList files, and they are Real PlayList and Virtual PlayList. "xxxxx.rpls" A file stores the information relevant to one Real PlayList. One file is made for every Real PlayList. A file name is "xxxxx.rpls." Here, "xxxxx" is a figure to five 0 thru/or 9. A file extension child presupposes that it must be "rpls".

[0120] A "yyyyy.vpls" file stores the information relevant to one Virtual PlayList. One file is made for every Virtual PlayList. A file name is "yyyyy.vpls." Here, "yyyyy" is a figure to five 0 thru/or 9. A file extension child presupposes that it must be "vpls".

[0121] A "CLIPINF" directory stores one file corresponding to each AV stream file. "zzzzz.clpi" A file is Clip Information file corresponding to one AV stream file (Clip AV stream file or Bridge-Clip AV stream file). A file name is "zzzzz.clpi" and "zzzzz" is a figure to five 0 thru/or 9. A file extension child presupposes that it must be "clpi".

[0122] An "M2TS" directory stores the file of AV stream. A "zzzzz.m2ts" file is

an AV stream file treated by the DVR system. This is Clip AV stream file or Bridge-Clip AV stream. A file name is "zzzzz.m2ts" and "zzzzz" is a figure to five 0 thru/or 9. A file extension child presupposes that it must be "m2ts."

[0123] The "DATA" directory stores the data transmitted from data broadcasting, and data are for example, XML file, an MHEG file, etc.

[0124] Next, the syntax and semantics of each directory (file) are explained. First, a "info.dvr" file is explained. Drawing 15 is drawing showing the syntax of a "info.dvr" file. A "info.dvr" file consists of three objects and they are DVRVolume(), TableOfPlayLists(), and MakerPrivateData().

[0125] TableOfPlayLists_Start_address shows the start address of TableOfPlayList() for explaining the syntax of info.dvr shown in drawing 15 by making the relative byte count from the cutting tool of the head of an info.dvr file into a unit. A relative byte count is counted from zero.

[0126] MakerPrivateData_Start_address shows the start address of MakerPrivateData() by making the relative byte count from the cutting tool of the head of an info.dvr file into a unit. A relative byte count is counted from zero. padding_word (padding WORD) is inserted according to the syntax of info.dvr. N1 and N2 are the positive integers of zero or arbitration. You may make it each padding WORD take any value.

[0127] DVRVolume() stores the information which describes the content of volume (disk). Drawing 16 is drawing showing the syntax of DVRVolume(). version_number shows four character alphabetic characters which show the version number of this DVRVolume() for explaining the syntax of DVR Volume() shown in drawing 16 . version_number is encoded with "0045" according to ISO 646.

[0128] length is expressed with the 32-bit unsigned integer which shows the byte count of DVRVolume() from immediately after this length field to the last of DVRVolume().

[0129] ResumeVolume() has memorized the file name of Real PlayList reproduced at the end in volume, or Virtual PlayList. However, a playback location when a user interrupts playback of Real PlayList or Virtual PlayList is stored in resume-mark defined in PlayListMark().

[0130] Drawing 17 is drawing showing the syntax of ResumeVolume(). For

explaining the syntax of ResumeVolume() shown in drawing 17 , valid_flag shows that the resume_PlayList_name field is invalid, when it is shown that the resume_PlayList_name field is effective when this 1-bit flag is set to 1 and this flag is set to 0.

[0131] 10 bytes of field of resume_PlayList_name shows the file name of Real PlayList by which resume should be carried out, or Virtual PlayList.

[0132] UIAppInfoVolume in the syntax of DVRVolume() shown in drawing 16 The parameter of the user interface application about volume is stored. The 8-bit field of character_set shows the coding approach of the character alphabetic character encoded in the Volume_name field for drawing 18 to be drawing showing the syntax of UIAppInfoVolume, and explain the semantics. The coding approach corresponds to the value shown in drawing 19 .

[0133] Eight bit fields of name_length show the cutting tool length of a volume name shown in the Volume_name field. The field of Volume_name shows the name of volume. The byte count of the left in this field to a name_length number is an effective character alphabetic character, and it shows the name of volume. In the Volume_name field, as for the value after these effective characters alphabetic character, what kind of value may be in close.

[0134] Volume_protect_flag is a flag which shows whether the contents in volume may be shown without restricting to a user. Only when this flag is set to 1 and a user is able to input an PIN number (password) correctly, showing a user the contents of that volume (reproduced) is permitted. When this flag is set to 0, even if a user does not input an PIN number, showing a user the contents of that volume is permitted.

[0135] If a user is able to input an PIN number correctly even if first this flag is set to 0 or this flag is set to 1, when a user inserts a disk in a player, as for the record regenerative apparatus 1, the list of PlayList in that disk will be displayed. The playback limit of each PlayList is unrelated to volume_protect_flag, and it is shown by playback_control_flag defined in UIAppInfoPlayList().

[0136] PIN consists of figures to four 0 thru/or 9, and each figure is encoded according to ISO/IEC 646. The field of ref_thumbnail_index shows the information on the thumbnail image added to volume. In the case of the value

whose ref_thumbnail_index field is not 0xFFFF, the thumbnail image is added to the volume and the thumbnail image is stored in the menu.thum file. The image is referred to using the value of ref_thumbnail_index in a menu.thum file. the ref_thumbnail_index field -- 0xFFFF it is -- a case -- the volume -- a thumbnail image -- adding -- having -- **** -- things -- being shown .

[0137] Next, TableOfPlayLists() in the syntax of info.dvr shown in drawing 15 is explained. TableOfPlayLists() stores the file name of PlayList (Real PlayList and Virtual PlayList). All the PlayList files currently recorded on volume are included in TableOfPlayList(). TableOfPlayLists() shows the default playback sequence of PlayList in volume.

[0138] version_number of TableOfPlayLists shows four character alphabetic characters which show the version number of this TableOfPlayLists for drawing 20 to be drawing showing the syntax of TableOfPlayLists(), and explain that syntax. version_number must be encoded with "0045" according to ISO 646.

[0139] length is an integer without a 32-bit sign which shows the byte count of TableOfPlayLists() from immediately after this length field to the last of TableOfPlayLists(). The 16-bit field of number_of_PlayLists shows the loop count of for-loop containing PlayList_file_name. This figure must be equal to the number of PlayList(s) currently recorded on volume. 10 bytes of figure of PlayList_file_name shows the file name of PlayList.

[0140] Drawing 21 is drawing showing the configuration of another operation of the syntax of TableOfPlayLists(). The syntax shown in drawing 21 is considered as the configuration in which UIAppinfoPlayList (after-mentioned) was included in the syntax shown in drawing 20 . Thus, it becomes possible only by reading TableOfPlayLists to create a menu screen by considering as the configuration in which UIAppinfoPlayList was included. Here, the following explanation is given noting that the syntax shown in drawing 20 is used.

[0141] MakersPrivateData in the syntax of info.dvr shown in drawing 15 is explained. MakersPrivateData is prepared so that the manufacturer of the record regenerative apparatus 1 can insert a manufacturer's private data into MakersPrivateData() for the special application of each company. Each manufacturer's private data has maker_ID standardized in order to identify the

manufacturer who defined it. MakersPrivateData() may also contain one or more maker_ID.

[0142] When a predetermined manufacturer wants to insert private data and other manufacturers' private data is already contained in MakersPrivateData(), other manufacturers add new private data into MakersPrivateData() rather than eliminate the old private data which already exists. Thus, in the gestalt of this operation, two or more manufacturers' private data carries out as [be / being contained in one MakersPrivateData() / possible].

[0143] Drawing 22 is drawing showing the syntax of MakersPrivateData. version_number shows four character alphabetic characters which show the version number of this MakersPrivateData() for explaining the syntax of MakersPrivateData shown in drawing 22 . version_number must be encoded with "0045" according to ISO 646. length shows the 32-bit unsigned integer which shows the byte count of MakersPrivateData() from immediately after this length field to the last of MakersPrivateData().

[0144] mpd_blocks_start_address shows the head byte address of the first mpd_block() by making the relative byte count from the cutting tool of the head of MakersPrivateData() into a unit. A relative byte count is counted from zero. number_of_maker_entries is a 16-bit unsigned integer which gives the number of entries of the manufacturer private data contained in MakersPrivateData(). Two or more manufacturer private data which have the value of the same maker_ID in MakersPrivateData() must not exist.

[0145] mpd_block_size is a 16-bit unsigned integer which gives the magnitude of one mpd_block by making 1024 bytes into a unit. For example, if it becomes mpd_block_size=1, it shows that the magnitude of one mpd_block is 1024 bytes. number_of_mpd_blocks is a 16-bit unsigned integer which gives the number of mpd_block contained in MakersPrivateData(). maker_ID is a 16-bit unsigned integer which shows the manufacture manufacturer of the DVR system which created the manufacturer private data. The value encoded by maker_ID is specified by the licenser of this DVR format.

[0146] maker_model_code is a 16-bit unsigned integer which shows the model number code of the DVR system which created the manufacturer private data. The value encoded by maker_model_code is set up by the

carrier beam manufacture manufacturer in the license of this format. start_mpd_block_number is a 16-bit unsigned integer which shows the number of mpd_block by which the manufacturer private data is started. The aryne of the initial data of manufacturer private data must be carried out to the head of mpd_block. start_mpd_block_number corresponds to the variable j in for-loop of mpd_block.

[0147] mpd_length is a 32-bit unsigned integer which shows the magnitude of manufacturer private data per cutting tool. mpd_block is a field in which manufacturer private data is stored. All mpd_block in MakersPrivateData() must be the same sizes.

[0148] Next, xxxxx.rpls and yyyyy.vpls will be explained if it puts in another way about Real PlayList file and Virtual PlayList file. Drawing 23 is drawing showing the syntax of xxxxx.rpls (Real PlayList) or yyyyy.vpls (Virtual PlayList). xxxxx.rpls and yyyyy.vpls have the same syntax configuration. xxxxx.rpls and yyyyy.vpls consist of three objects, respectively, and they are PlayList(), PlayListMark(), and MakerPrivateData().

[0149] PlayListMark_Start_address shows the start address of PlayListMark() by making the relative byte count from the cutting tool of the head of a PlayList file into a unit. A relative byte count is counted from zero.

[0150] MakerPrivateData_Start_address shows the start address of MakerPrivateData() by making the relative byte count from the cutting tool of the head of a PlayList file into a unit. A relative byte count is counted from zero.

[0151] padding_word (padding WORD) is inserted according to the syntax of a PlayList file, and N1 and N2 are the positive integers of zero or arbitration. You may make it each padding WORD take any value.

[0152] Here, although already explained simple, PlayList is explained further. Refer to the playback section in all Clip(s) except Bridge-Clip (after-mentioned) for all Real PlayList in a disk. And two or more RealPlayList(s) must not make the playback section shown by those PlayItem(s) overlap in the same Clip.

[0153] As shown, Real PlayList to which all Clip(s) correspond exists in explaining further with reference to drawing 24 at drawing 24 (A). This

regulation is followed after an editing task is performed, as shown in drawing 24 (B). therefore, all Clip(s) -- which -- it is -- surely viewing and listening is possible by referring to Real PlayList.

[0154] As shown in drawing 24 (C), the playback section of Virtual PlayList must be included in the playback section of Real PlayList, or the playback section of Bridge-Clip. Bridge-Clip referred to at no Virtual PlayList must not exist in a disk.

[0155] Although RealPlayList includes the list of PlayItem, it must not contain SubPlayItem. When CPI_type Virtual PlayList is indicated to be in PlayList() including the list of PlayItem is EP_map type and PlayList_type is 0 (PlayList containing video and an audio), Virtual PlayList can contain one SubPlayItem. In PlayList() in the gestalt of this operation, SubPlayIte must be used only for the object of postrecording of an audio and the number of SubPlayItem(s) which one Virtual PlayList has must be 0 or 1.

[0156] Next, PlayList is explained. Drawing 25 is drawing showing the syntax of PlayList. They are four character alphabetic characters in which version_number shows the version number of this PlayList() for explaining the syntax of PlayList shown in drawing 25 . version_number must be encoded with "0045" according to ISO 646. length is a 32-bit unsigned integer which shows the byte count of PlayList() from immediately after this length field to the last of PlayList(). PlayList_type is the 8-bit field which shows the type of this PlayList, and shows that example to drawing 26 .

[0157] CPI_type is a 1-bit flag and shows the value of CPI_type of Clip referred to by PlayItem() and SubPlayItem(). All Clip(s) referred to by one PlayList must have the the same value of CPI_type defined in those CPI(). number_of_PlayItems is the 16-bit field which shows the number of PlayItem(s) in PlayList.

[0158] PlayItem_id corresponding to predetermined PlayItem() is defined in for-loop containing PlayItem() by the sequence that the PlayItem() appears. PlayItem_id is started from 0. number_of_SubPlayItems is the 16-bit field which shows the number of SubPlayItem(s) in PlayList. This value is 0 or 1. The pass (audio stream pass) of an additional audio stream is a kind of subpass.

[0159] Next, UIAppInfoPlayList of the syntax of PlayList shown in drawing 25 is explained. UIAppInfoPlayList stores the parameter of the user interface application about PlayList. Drawing 27 is drawing showing the syntax of UIAppInfoPlayList. For explaining the syntax of UIAppInfoPlayList shown in drawing 27 , character_set is the 8-bit field and shows the coding approach of the character alphabetic character encoded in the PlayList_name field. The coding approach corresponds to the value based on the table shown in drawing 19 .

[0160] name_length is eight bit fields and shows the cutting tool length of the PlayList name shown in the PlayList_name field. The field of PlayList_name shows the name of PlayList. The byte count of the left in this field to a name_length number is an effective character alphabetic character, and it shows the name of PlayList. In the PlayList_name field, as for the value after these effective characters alphabetic character, what kind of value may be in close.

[0161] record_time_and_date is the 56-bit field in which time when PlayList is recorded is stored. This field encodes 14 figures by 4-bit Binary Coded Decimal (BCD) about a /part / second at the time of year / moon / day/. For example, 2001/12/23:01:02:03 It encodes with "0x20011223010203."

[0162] duration is the 24-bit field which showed the total playback time amount of PlayList in the unit of time amount / part / second. This field encodes six figures by 4-bit Binary CodedDecimal (BCD). For example, 01:45:30 is encoded with "0x014530."

[0163] valid_period is the 32-bit field which shows the period when PlayList is effective. This field encodes eight figures by 4-bit Binary Coded Decimal (BCD). For example, the record regenerative apparatus 1 is used as it said that automatic elimination of the PlayList over which this shelf-life passed was carried out. For example, 2001/05/07 It encodes with "0x20010507."

[0164] maker_id is a 16-bit unsigned integer which shows the manufacturer of the DVR player (record regenerative apparatus 1) which updated the PlayList at the end. The value encoded by maker_id is assigned by the licenser of a DVR format. maker_code is a 16-bit unsigned integer which shows the model number of the DVR player which updated the PlayList at the end. The value

encoded by maker_code is decided by the carrier beam manufacturer in the license of a DVR format.

[0165] The PlayList is reproduced, only when the flag of playback_control_flag is set to 1 and a user is able to input an PIN number correctly. When this flag is set to 0, even if a user does not input an PIN number, a user can view and listen to that PlayList.

[0166] As a table is shown in drawing 28 (A), when write_protect_flag is set to 1, write_protect_flag is removed, and the content of the PlayList is not eliminated and changed. When this flag is set to 0, a user can eliminate and change that PlayList freely. When this flag is set to 1, before a user eliminates, edits or overwrites that PlayList, the record regenerative apparatus 1 displays a message which is reconfirmed to a user.

[0167] Virtual PlayList which Real PlayList by which write_protect_flag is set to 0 exists, and refers to Clip of the Real PlayList exists, and write_protect_flag of the Virtual PlayList may be set to 1. "Minimize" [regenerative apparatus / it warns a user of existence of Above Virtual PlayList, or / the Real PlayList] before the record regenerative apparatus 1 eliminates the Real PlayList when a user is going to eliminate RealPlayList.

[0168] As is_played_flag is shown in drawing 28 (B), when it is shown that it was reproduced at once after the PlayList was recorded when the flag was set to 1 and it is set to 0, the PlayList shows not being reproduced once, after being recorded.

[0169] archive is the 2-bit field which shows whether the PlayList is original or it is copied, as shown in drawing 28 (C). ref_thumbnail_index The field shows the information on a thumbnail image that PlayList is represented. In the case of the value whose ref_thumbnail_index field is not 0xFFFF, the thumbnail image representing PlayList is added to the PlayList, and the thumbnail image is menu.thum. It is stored in the file. The image is referred to using the value of ref_thumbnail_index in a menu.thum file. the ref_thumbnail_index field -- 0xFFFF it is -- a case -- the PlayList -- PlayList -- representing -- a thumbnail -- an image -- adding -- having -- **** .

[0170] Next, PlayItem is explained. One PlayItem() contains the following data fundamentally. When CPI_type defined in the pair of IN_time for specifying

Clip_information_file_name for specifying the file name of Clip and the playback section of Clip and OUT_time and PlayList() is EP_map type, they are STC_sequence_id which IN_time and OUT_time refer to, and connection_condition which shows the condition of connection between PlayItem to precede and current PlayItem.

[0171] Those PlayItem(s) are arranged in on the global time-axis of PlayList without the gap of time amount, or overlap by the single tier when PlayList consists of two or more PlayItem(s). CPI_type defined in PlayList() is EP_map type, and the pair of IN_time defined in the PlayItem when the present PlayItem does not have BridgeSequence(), and OUT_time must point out the time amount on the same STC continuation section specified by STC_sequence_id. Such an example is shown in drawing 29 .

[0172] CPI_type defined in PlayList() is EP_map type, and drawing 30 shows the case where the regulation explained below is applied, when the present PlayItem has BridgeSequence(). IN_time (what is indicated to be IN_time1 in drawing) of PlayItem preceded with current PlayItem has pointed out the time amount on the STC continuation section specified by STC_sequence_id of PlayItem to precede. OUT_time (what is indicated to be OUT_time1 in drawing) of PlayItem to precede has pointed out the time amount in Bridge-Clip specified in BridgeSequenceInfo() of current PlayItem. This OUT_time must follow the coding limit mentioned later.

[0173] IN_time (what is indicated to be IN_time2 in drawing) of current PlayItem has pointed out the time amount in Bridge-Clip specified in BridgeSequenceInfo() of current PlayItem. This IN_time must also follow the coding limit mentioned later. OUT_time (what is indicated to be OUT_time2 in drawing) of PlayItem of current PlayItem has pointed out the time amount on the STC continuation section specified by STC_sequence_id of current PlayItem.

[0174] As shown in drawing 31 , when CPI_type of PlayList() is TU_map type, the pair of IN_time of PlayItem and OUT_time has pointed out the time amount on the same Clip AV stream.

[0175] The syntax of PlayItem comes to be shown in drawing 32 . The field of Clip_Information_file_name shows the file name of ClipInformation file for

explaining the syntax of PlayItem shown in drawing 32 . Clip_stream_type defined in ClipInfo() of this Clip Information file must show Clip AV stream.
[0176] STC_sequence_id is the 8-bit field and shows STC_sequence_id of the STC continuation section which PlayItem refers to. When CPI_type specified in PlayList() is TU_map type, these eight bit fields have no semantics, but are set to 0. IN_time is 32 bit fields and stores the playback start time of PlayItem. The semantics of IN_time changes with CPI_type defined in PlayList(), as shown in drawing 33 .
[0177] OUT_time is 32 bit fields and stores the playback end time of PlayItem. The semantics of OUT_time changes with CPI_type defined in PlayList(), as shown in drawing 34 .
[0178] Connection_Condition is the 2-bit field which shows the connection condition between PlayItem to precede as shown in drawing 35 , and the present PlayItem. Drawing 36 is drawing explaining each condition of Connection_Condition shown in drawing 35 .
[0179] Next, BridgeSequenceInfo is explained with reference to drawing 37 . BridgeSequenceInfo() is the attached information on current PlayItem, and has the information shown below. Bridge_Clip_Information_file_name which specifies Clip Information file corresponding to a Bridge-Clip AV stream file and it is included.
[0180] Moreover, it is the address of the source packet on Clip AV stream which PlayItem to precede refers to, and the source packet of the beginning of a Bridge-Clip AV stream file is connected following this source packet. This address is called RSPN_exit_from_previous_Clip. It is the address of the source packet on Clip AV stream which further current PlayItem refers to, and the source packet of the last of a Bridge-Clip AV stream file is connected before this source packet. This address is called RSPN_enter_to_current_Clip.
[0181] In drawing 37 , RSPN_arrival_time_discontinuity shows the address of the source packet which has the break point of arrival time base in a the Bridge-Clip AVstream file. This address is defined in ClipInfo().
[0182] Drawing 38 is drawing showing the syntax of BridgeSequenceinfo. The field of Bridge_Clip_Information_file_name shows the file name of Clip Information file corresponding to a Bridge-Clip AV stream file for explaining

the syntax of BridgeSequenceinfo shown in drawing 38 . Clip_stream_type defined in ClipInfo() of this Clip Information file must show 'Bridge-Clip AV stream'.

[0183] 32 bit fields of RSPN_exit_from_previous_Clip are the relative addresses of the source packet on Clip AV stream which PlayItem to precede refers to, and the source packet of the beginning of a Bridge-Clip AV stream file is connected following this source packet. RSPN_exit_from_previous_Clip is magnitude which makes a source packet number a unit, and counts the value of offset_SPN defined in ClipInfo() from the source packet of the beginning of the Clip AV stream file which PlayItem to precede refers to as initial value.

[0184] 32 bit fields of RSPN_enter_to_current_Clip are the relative addresses of the source packet on Clip AV stream which current PlayItem refers to, and the source packet of the last of a Bridge-Clip AV stream file is connected before this source packet. RSPN_exit_from_previous_Clip is magnitude which makes a source packet number a unit, and counts the value of offset_SPN defined in ClipInfo() from the source packet of the beginning of the Clip AV stream file which current PlayItem refers to as initial value.

[0185] Next, SubPlayItem is explained with reference to drawing 39 . The activity of SubPlayItem() is allowed only when CPI_type of PlayList() is EP_map type. In the gestalt of this operation, SubPlayItem presupposes that it is used only for the object of postrecording of an audio. SubPlayItem() contains the data shown below. First, Clip_information_file_name for specifying Clip which sub path in PlayList refers to is included.

[0186] Moreover, SubPath_IN_time for specifying the playback section of sub path in Clip SubPath_OUT_time is included. Furthermore, sync_PlayItem_id for specifying the time of day in which sub path carries out playback initiation on the time-axis of main path sync_start_PTS_of_PlayItem is included. Clip AV stream of the audio referred to at sub path must not contain an STC break point (break point of system time base). The clock of the audio sample of Clip used for sub path is locked by the clock of the audio sample of main path.

[0187] Drawing 40 is drawing showing the syntax of SubPlayItem. The field of Clip_Information_file_name shows the file name of Clip Information file for

explaining the syntax of SubPlayItem shown in drawing 40 , and it is used by sub path in PlayList. Clip_stream_type defined in ClipInfo() of this Clip Information file must show Clip AV stream.

[0188] The 8-bit field of SubPath_type shows the type of sub path. Here, as shown in drawing 41 , only '0x00' is set up but other values are secured for the future.

[0189] The 8-bit field of sync_PlayItem_id shows PlayItem_id of PlayItem in which the time of day sub path carries out [time of day] playback initiation on the time-axis of main path is contained. The value of PlayItem_id corresponding to predetermined PlayItem is defined in PlayList() (refer to drawing 25).

[0190] The 32-bit field of sync_start_PTS_of_PlayItem shows the time of day in which sub path carries out playback initiation on the time-axis of main path, and shows 32 bits of high orders of PTS on PlayItem referred to by sync_PlayItem_id (Presentaiotn Time Stamp). 32 bit fields of SubPath_IN_time store the playback start time of Sub path. SubPath_IN_time shows 32 bits of high orders of PTS of 33 bit length corresponding to the first presentation unit in Sub Path.

[0191] 32 bit fields of SubPath_OUT_time store the playback end time of Sub path. SubPath_OUT_time shows 32 bits of high orders of the value of Presenation_end_TS computed by the degree type. Presentation_end_TS = PTS_out+AU_duration -- here, PTS_out is PTS of 33 bit length corresponding to the presentation unit of the last of SubPath. AU_duration is the display period of the 90kHz unit of the presentation unit of the last of SubPath.

[0192] Next, PlayListMark() in the syntax of xxxxx.rpls shown in drawing 23 and yyyyy.vpls is explained. The mark information about PlayList is stored in this PlayListMark. Drawing 42 is drawing showing the syntax of PlayListMark. They are four character alphabetic characters in which version_number shows the version number of this PlayListMark() for explaining the syntax of PlayListMark shown in drawing 42 . version_number must be encoded with "0045" according to ISO 646.

[0193] length is a 32-bit unsigned integer which shows the byte count of PlayListMark() from immediately after this length field to the last of

PlayListMark(). number_of_PlayList_marks is a 16-bit unsigned integer which shows the number of the mark currently stored in PlayListMark. number_of_PlayList_marks You may be 0. mark_type is the 8-bit field which shows the type of a mark, and is encoded according to the table shown in drawing 43 .

[0194] 32 bit fields of mark_time_stamp store the time stump in which the point with which the mark was specified is shown. The semantics of mark_time_stamp changes with CPI_type defined in PlayList(), as shown in drawing 44 . PlayItem_id is the 8-bit field which specifies PlayItem on which the mark is put. The value of PlayItem_id corresponding to predetermined PlayItem is defined in PlayList() (refer to drawing 25).

[0195] The 8-bit field of character_set shows the coding approach of the character alphabetic character encoded by the mark_name field. The coding approach corresponds to the value shown in drawing 19 . Eight bit fields of name_length show the cutting tool length of the mark name shown in the Mark_name field. The field of mark_name shows the name of a mark. The byte count of the left in this field to a name_length number is an effective character alphabetic character, and it shows the name of a mark. As for the value after these effective characters alphabetic character, what kind of value may be set up in the Mark_name field.

[0196] The field of ref_thumbnail_index shows the information on the thumbnail image added to a mark. In the case of the value whose ref_thumbnail_index field is not 0xFFFF, the thumbnail image is added to the mark and the thumbnail image is stored in the mark.thmb file. The image is referred to using the value of ref_thumbnail_index in a mark.thmb file (after-mentioned). the ref_thumbnail_index field -- 0xFFFF it is -- a case -- the mark -- a thumbnail image -- adding -- having -- **** -- things -- being shown .

[0197] Next, Clip information file is explained. zzzzz.clpi (Clip information file file) consists of six objects, as shown in drawing 45 . They are ClipInfo(), STC_Info(), ProgramInfo(), CPI(), ClipMark(), and MakerPrivateData(). "zzzzz" of the digit string with the same Clip Information file corresponding to AV stream (a Clip AV stream or Bridge-Clip AV stream) and it is used.

[0198] ClipInfo_Start_address shows the start address of ClipInfo() for

explaining the syntax of zzzzz.clpi (Clip information file file) shown in drawing 45 by making the relative byte count from the cutting tool of the head of a zzzzz.clpi file into a unit. A relative byte count is counted from zero.

[0199] STC_Info_Start_address shows the start address of STC_Info() by making the relative byte count from the cutting tool of the head of a zzzzz.clpi file into a unit. A relative byte count is counted from zero. ProgramInfo_Start_address shows the start address of ProgramInfo() by making the relative byte count from the cutting tool of the head of a zzzzz.clpi file into a unit. A relative byte count is counted from zero. CPI_Start_address shows the start address of CPI() by making the relative byte count from the cutting tool of the head of a zzzzz.clpi file into a unit. A relative byte count is counted from zero.

[0200] ClipMark_Start_address shows the start address of ClipMark() by making the relative byte count from the cutting tool of the head of a zzzzz.clpi file into a unit. A relative byte count is counted from zero. MakerPrivateData_Start_address shows the start address of MakerPrivateData () by making the relative byte count from the cutting tool of the head of a zzzzz.clpi file into a unit. A relative byte count is counted from zero. padding_word (padding WORD) is inserted according to the syntax of a zzzzz.clpi file. N1, N2, N3, N4, and N5 must be the positive integers of zero or arbitration. As for each padding WORD, any value may be made to be taken.

[0201] Next, ClipInfo is explained. Drawing 46 is drawing showing the syntax of ClipInfo. ClipInfo() stores the attribute information on AV stream file (a Clip AV stream or Bridge-Clip AV stream file) corresponding to it.

[0202] They are four character alphabetic characters in which version_number shows the version number of this ClipInfo() for explaining the syntax of ClipInfo shown in drawing 46 . version_number must be encoded with "0045" according to ISO 646. length is a 32-bit unsigned integer which shows the byte count of ClipInfo() from immediately after this length field to the last of ClipInfo(). The 8-bit field of Clip_stream_type shows the type of AV stream corresponding to a Clip Information file, as shown in drawing 47 . About the stream type of each type of AV stream, it mentions later.

[0203] The 32-bit field of offset_SPN gives the offset value of the source

packet number about the source packet of the beginning of AV stream (Clip AV stream or Bridge-Clip AV stream) file. This offset_SPN must be 0 when AV stream file is first recorded on a disk.

[0204] As shown in drawing 48 , when the first part of AV stream file is eliminated by edit, offset_SPN is very good in values other than zero. The relative source packet number (relative address) which refers to offset_SPN with the gestalt of this operation is often RSPN_xxx (xxx deforms.). It is described by in the form of example .RSPN_EP_start in syntax. A relative source packet number is magnitude which makes a source packet number a unit, and counts the value of offset_SPN as initial value from the source packet of the beginning of AV stream file.

[0205] The number (SPN_xxx) of the source packets to the source packet referred to by the relative source packet number from the source packet of the beginning of AV stream file is computed by the degree type.

An example in case offset_SPN is 4 is shown in SPN_xxx = RSPN_xxx-offset_SPN drawing 48 .

[0206] TS_recording_rate -- a 24-bit unsigned integer -- it is -- this value -- a DVR drive (write-in section 22) -- or the bit rate of required I/O of AV stream from a DVR drive (read-out section 28) is given. record_time_and_date is the 56-bit field in which time when AV stream corresponding to Clip is recorded is stored, and encodes 14 figures by 4-bit Binary Coded Decimal (BCD) about a /part / second at the time of year / moon / day/. For example, 2001/12/23:01:02:03 are encoded with "0x20011223010203."

[0207] duration is the 24-bit field which showed the total playback time amount of Clip in the unit of time amount / part / second based on an arrival timer clock. This field encodes six figures by 4-bit Binary Coded Decimal (BCD). For example, 01:45:30 is encoded with "0x014530."

[0208] The flag of time_controlled_flag: shows the recording mode of AV stream file. When this time_controlled_flag is 1, a recording mode must fulfill the conditions which show that it is the mode in which it is recorded to the time amount progress after recording as a file size is proportional, and are shown in a degree type.

TS_average_rate*192/188*(t-start_time)-alpha <= -- size_clip(t) <=

TS_average_rate*192 / 188*(t-start_time)+alpha -- here -- TS_average_rate -- the average bit rate of the transport stream of AV stream file -- bytes/second a unit -- a table -- it is a thing the bottom.

[0209] Moreover, in a top type, t shows the time amount expressed per second, and start_time is time of day when the source packet of the beginning of AV stream file is recorded, and is expressed per second. size_clip (t), When the size of AV stream file in time of day t is expressed per cutting tool, for example, ten source packets are recorded by time of day t from start_time, size_clip (t) is 10*192 bytes. alpha is a constant depending on TS_average_rate.

[0210] When time_controlled_flag is set to 0, not controlling the recording mode so that the file size of AV stream is proportional to time amount progress of record is shown. For example, this is the case where transparent record of the input transport stream is carried out.

[0211] When, as for TS_average_rate, time_controlled_flag is set to 1, this 24-bit field shows the value of TS_average_rate used by the top formula. When time_controlled_flag is set to 0, this field has no semantics but must be set to 0. For example, the transport stream of a Variable Bit Rate is encoded by the procedure shown below. A transformer portrait is first set to the value of TS_recording_rate. Next, a video stream is encoded with a Variable Bit Rate. And transport Paquette is intermittently encoded by not using Nur Paquette.

[0212] 32 bit fields of RSPN_arrival_time_discontinuity are the relative addresses of the location which the discontinuity of arrival time base generates on a Bridge-Clip AV stream file. RSPN_arrival_time_discontinuity is magnitude which makes a source packet number a unit, and is ClipInfo() from the source packet of the beginning of a Bridge-Clip AV stream file. The value of offset_SPN set and defined is counted as initial value. The absolute address in the inside of the Bridge-Clip AV stream file is computed based on SPN_xxx = RSPN_xxx-offset_SPN mentioned above.

[0213] The 144-bit field of reserved_for_system_use is reserved for systems. When the flag of is_format_identifier_valid is 1, it is shown that the field of format_identifier is effective. When the flag of is_original_network_ID_valid is 1, it is shown that the field of original_network_ID is effective. When the flag of

is_transport_stream_ID_valid is 1, it is shown that the field of transport_stream_ID is effective. When the flag of is_servece_ID_valid is 1, it is shown that the field of servece_ID is effective.

[0214] When the flag of is_country_code_valid is 1, it is shown that the field of country_code is effective. 32 bit fields of format_identifier show the value of format_identifier which registration deascriotor (it defines as ISO/IEC 13818-1) has in a transport stream. 16 bit fields of original_network_ID show the value of original_network_ID defined in the transport stream. 16 bit fields of transport_stream_ID show the value of transport_stream_ID defined in the transport stream.

[0215] 16 bit fields of servece_ID show the value of servece_ID defined in the transport stream. The 24-bit field of country_code shows the country code defined by ISO3166. Each character alphabetic character is encoded by ISO 8859-1. For example, Japan is expressed as "JPN" and encoded with "0x4A 0x500x4E." stream_format_name is 16 character codes of ISO-646 which show the name of the format engine which is doing the stream definition of a transport stream. As for the invalid cutting tool in this field, value'0xFF' is set.

[0216] format_identifier, original_network_ID, transport_stream_ID, servece_ID, country_code, and stream_format_name can show the service provider of a transport stream, and, thereby, can recognize a coding limit of an audio or a video stream, and the stream definition of the specification of SI (service information), or private data streams other than an audio video stream. Such information can be used, in order that a decoder may perform initial setting of a decoder system before decoding initiation whether the stream can be decoded and when it can decode and.

[0217] Next, STC_Info is explained. Here, the time amount section which does not contain the break point (break point of system time base) of STC in an MPEG-2 transport stream is called STC_sequence, and STC_sequence is specified with the value of STC_sequence_id in Clip. Drawing 50 is drawing explaining the STC section [****]. The value of the STC same in the same STC_sequence never appears (however, the maximum time amount length of Clip is restricted so that it may mention later). Therefore, the same value of PTS also never appears in the same STC_sequence. When AV stream

contains the STC break point of N (N> 0) individual, the system time base of Clip is divided into STC_sequence of an individual (N+1).

[0218] STC_Info stores the address of the location which the discontinuity (discontinuity of system time base) of STC generates. RSPN_STC_start shows the address, and k-th STC_sequence (k>=0) except the last STC_sequence begins from the time of day when the source packet referred to by k-th RSPN_STC_start arrived, and finishes with the time of day when the source packet referred to by RSPN_STC_start of eye watch (k+1) arrived so that it may explain with reference to drawing 51 . The last STC_sequence begins from the time of day when the source packet referred to by the last RSPN_STC_start arrived, and is ended at the time of day when the last source packet arrived.

[0219] Drawing 52 is drawing showing the syntax of STC_Info. They are four character alphabetic characters in which version_number shows the version number of this STC_Info() for explaining the syntax of STC_Info shown in drawing 52 . version_number must be encoded with "0045" according to ISO 646.

[0220] length is a 32-bit unsigned integer which shows the byte count of STC_Info() from immediately after this length field to the last of STC_Info(). When CPI_type of CPI() shows TU_map type, this length field may set zero. When CPI_type of CPI() shows EP_map type, num_of_STC_sequences must be one or more values.

[0221] The 8-bit unsigned integer of num_of_STC_sequences shows the number of STC_sequence(s) in the inside of Clip. This value shows the loop count of for-loop following this field. STC_sequence_id corresponding to predetermined STC_sequence is defined in for-loop containing RSPN_STC_start by the sequence that RSPN_STC_start corresponding to the STC_sequence appears. STC_sequence_id is started from 0.

[0222] 32 bit fields of RSPN_STC_start show the address which STC_sequence starts on AV stream file. RSPN_STC_start shows the address which the break point of system time base generates in AV stream file. RSPN_STC_start is good also as a relative address of a source packet which has PCR of the beginning of new system time base in AV stream.

RSPN_STC_start is magnitude which makes a source packet number a unit, and counts the value of offset_SPN defined in ClipInfo() from the source packet of the beginning of AV stream file as initial value. The absolute address in the inside of the AV stream file is computed by SPN_xxx = RSPN_xxx-offset_SPN already mentioned above.

[0223] Next, ProgramInfo in the syntax of zzzzz.clip shown in drawing 45 is explained. The time amount section which has the following description in Clip is called program_sequence for explaining here, referring to drawing 53 . First, the value of PCR_PID does not change. Next, the number of video elementary streams does not change. Moreover, the encoded information defined by the value and VideoCodingInfo of PID about each video stream does not change. Furthermore, the number of audio elementary streams does not change. Moreover, the encoded information defined by the value and AudioCodingInfo of PID about each audio stream does not change.

[0224] program_sequence has only one system time base in the same time of day. program_sequence has only one PMT in the same time of day. ProgramInfo() stores the address of the location which program_sequence starts. RSPN_program_sequence_start shows the address.

[0225] Drawing 54 is drawing showing the syntax of ProgramInfo. They are four character alphabetic characters in which version_number shows the version number of this ProgramInfo() for explaining SHINTAKU of ProgramInfo shown in drawing 54 . version_number must be encoded with "0045" according to ISO 646.

[0226] length is a 32-bit unsigned integer which shows the byte count of ProgramInfo() from immediately after this length field to the last of ProgramInfo(). When CPI_type of CPI() shows TU_map type, this length field may be set to zero. When CPI_type of CPI() shows EP_map type, number_of_programs must be one or more values.

[0227] The 8-bit unsigned integer of number_of_program_sequences shows the number of program_sequence in the inside of Clip. This value shows the loop count of for-loop following this field. When program_sequence does not change in Clip, number_of_program_sequences must have 1 set. 32 bit fields of RSPN_program_sequence_start are the relative addresses of the location

which a program sequence starts on AV stream file.

[0228] RSPN_program_sequence_start is magnitude which makes a source packet number a unit, and counts the value of offset_SPN defined in ClipInfo() from the source packet of the beginning of AV stream file as initial value. The absolute address in the inside of the AV stream file is computed by SPN_xxx = RSPN_xxx-offset_SPN. A RSPN_program_sequence_start value must appear in ascending order in for-loop of syntax.

[0229] 16 bit fields of PCR_PID show PID of transport Paquette including the PCR field effective in the program_sequence. Eight bit fields of number_of_videos show the loop count of video_stream_PID and for-loop containing VideoCodingInfo(). Eight bit fields of number_of_audios show the loop count of audio_stream_PID and for-loop containing AudioCodingInfo(). 16 bit fields of video_stream_PID show PID of transport Paquette containing a video stream effective in the program_sequence. VideoCodingInfo() following this field must explain the content of the video stream referred to by that video_stream_PID.

[0230] 16 bit fields of audio_stream_PID show PID of transport Paquette containing an audio stream effective in the program_sequence. AudioCodingInfo() following this field must explain the content of the video stream referred to by that audio_stream_PID.

[0231] In addition, the sequence that the value of video_stream_PID appears in for-loop of syntax must be equal to the sequence that PID of a video stream is encoded in PMT effective in the program_sequence. Moreover, the sequence that the value of audio_stream_PID appears in for-loop of syntax must be equal to the sequence that PID of an audio stream is encoded in PMT effective in the program_sequence.

[0232] Drawing 55 is drawing showing the syntax of VideoCodingInfo in the syntax of Programinfo shown in drawing 54 . For explaining the syntax of VideoCodingInfo shown in drawing 55 , eight bit fields of video_format show the video format corresponding to video_stream_PID in ProgramInfo(), as shown at drawing 56 .

[0233] Eight bit fields of frame_rate show the frame rate of the video corresponding to video_stream_PID in ProgramInfo(), as shown in drawing

57 . Eight bit fields of display_aspect_ratio show the display aspect ratio of the video corresponding to video_stream_PID in ProgramInfo(), as shown in drawing 58 .

[0234] Drawing 59 is drawing showing the syntax of AudioCodingInfo in the syntax of Programinfo shown in drawing 54 . For explaining the syntax of AudioCodingInfo shown in drawing 59 , eight bit fields of audio_coding show the coding approach of the audio corresponding to audio_stream_PID in ProgramInfo(), as shown at drawing 60 .

[0235] Eight bit fields of audio_component_type show the component type of the audio corresponding to audio_stream_PID in ProgramInfo(), as shown in drawing 61 . Eight bit fields of sampling_frequency show the sampling frequency of the audio corresponding to audio_stream_PID in ProgramInfo(), as shown in drawing 62 .

[0236] Next, CPI in the syntax of zzzzz.clip shown in drawing 45 (Characteristic Point Information) is explained. Since the hour entry in AV stream and the address in the file are associated, there is CPI. There are two types of CPI(s) and they are EP_map and TU_map. As shown in drawing 63 , when CPI_type in CPI() is EP_map type, the CPI() contains EP_map. As shown in drawing 64 , when CPI_type in CPI() is TU_map type, the CPI() contains TU_map. One AV stream has one EP_map or one TU_map. When AV stream is a SESF transport stream, Clip corresponding to it must have EP_map.

[0237] Drawing 65 is drawing showing the syntax of CPI. They are four character alphabetic characters in which version_number shows the version number of this CPI() for explaining the syntax of CPI shown in drawing 65 . version_number must be encoded with "0045" according to ISO 646. length is a 32-bit unsigned integer which shows the byte count of CPI() from immediately after this length field to the last of CPI(). As shown in drawing 66 , CPI_type is a 1-bit flag and expresses the type of CPI of Clip.

[0238] Next, EP_map in the syntax of CPI shown in drawing 65 is explained. There are two types of EP_map and they are EP_map for video streams, and EP_map for audio streams. EP_map_type in EP_map distinguishes the type of EP_map. When Clip contains one or more video streams, EP_map for

video streams must be used. When Clip contains one or more audio streams excluding a video stream, EP_map for audio streams must be used.

[0239] EP_map for video streams is explained with reference to drawing 67 . EP_map for video streams has data called stream_PID, PTS_EP_start, and RSPN_EP_start. stream_PID shows PID of transport Paquette who transmits a video stream. PTS_EP_start shows PTS of the access unit begun from the sequence header of a video stream. RSPN_EP_start shows the address of the source pocket containing the 1st byte of the access unit referred to by PTS_EP_start in AV stream.

[0240] The sub table called EP_map_for_one_stream_PID() is made for every video stream transmitted by transport Paquette with the same PID. When two or more video streams exist in Clip, EP_map may also contain two or more EP_map_for_one_stream_PID().

[0241] EP_map for audio streams has data called stream_PID, PTS_EP_start, and RSPN_EP_start. stream_PID shows PID of transport Paquette who transmits an audio stream. PTS_EP_start shows PTS of the access unit of an audio stream. RSPN_EP_start shows the address of the source pocket containing the 1st byte of the access unit referred to by PTS_EP_start in AV stream.

[0242] The sub table called EP_map_for_one_stream_PID() is made for every audio stream transmitted by transport Paquette with the same PID. When two or more audio streams exist in Clip, EP_map may also contain two or more EP_map_for_one_stream_PID().

[0243] One EP_map_for_one_stream_PID() is made by explaining the relation between EP_map and STC_Info regardless of the break point of STC at one table. By comparing the value of RSPN_STC_start defined in the value of RSPN_EP_start and STC_Info() shows the boundary of the data of EP_map belonging to each STC_sequence (see drawing 68). EP_map must have one EP_map_for_one_stream_PID to the range of the continuous stream transmitted by the same PID. When shown in drawing 69 , although program#1 and program#3 have the same video PID, since the data range is not continuing, they must have EP_map_for_one_stream_PID for every program.

[0244] Drawing 70 is drawing showing the syntax of EP_map. For explaining the syntax of EP_map shown in drawing 70 , EP_type is the 4-bit field, and as shown at drawing 71 , it shows the entry point type of EP_map. EP_type shows the semantics of the data field following this field. EP_type must be set to 0 ('video') when Clip contains one or more video streams. Or EP_type must be set to 1 ('audio') when Clip contains one or more audio streams excluding a video stream.

[0245] The 16-bit field of number_of_stream_PIDs shows the loop count of for-loop which has number_of_stream_PIDs in EP_map() in a variable. The 16-bit field of stream_PID (k) shows PID of transport Paquette who transmits the k-th elementary stream (video or audio stream) referred to by EP_map_for_one_stream_PID (num_EP_entries (k)). case EP_type is equal to 0 ('video') -- the elementary stream -- a video stream -- kicking does not become impossible Moreover, when EP_type is equal to 1 ('audio'), the elementary stream must be an audio stream.

[0246] The 16-bit field of num_EP_entries (k) shows num_EP_entries (k) referred to by EP_map_for_one_stream_PID (num_EP_entries (k)). EP_map_for_one_stream_PID_Start_address (k): This 32-bit field shows the relative byte position from which EP_map_for_one_stream_PID (num_EP_entries (k)) begins in EP_map(). This value is shown by the magnitude from the 1st byte of EP_map().

[0247] padding_word must be inserted according to the syntax of EP_map(). X and Y must be the positive integers of zero or arbitration. Each padding WORD may take any value.

[0248] Drawing 72 is drawing showing the syntax of EP_map_for_one_stream_PID. The semantics of the 32-bit field of PTS_EP_start changes with EP_type(s) defined in EP_map() to explain the syntax of EP_map_for_one_stream_PID shown in drawing 72 . When EP_type is equal to 0 ('video'), this field has 32 bits of high orders of PTS of the 33-bit precision of the access unit which starts in the sequence header of a video stream. When EP_type is equal to 1 ('audio'), this field has 32 bits of high orders of PTS of the 33-bit precision of the access unit of an audio stream.

[0249] The semantics of the 32-bit field of RSPN_EP_start changes with

EP_type defined in EP_map(). When EP_type is equal to 0 ('video'), this field shows the relative address of the source pocket containing the 1st byte of the sequence header of the access unit referred to by PTS_EP_start in AV stream. Or when EP_type is equal to 1 ('audio'), this field shows the relative address of the source pocket containing the first byte of the audio frame of the access unit referred to by PTS_EP_start in AV stream.

[0250] RSPN_EP_start is magnitude which makes a source packet number a unit, and counts the value of offset_SPN defined in ClipInfo() from the source packet of the beginning of AV stream file as initial value. The absolute address in the inside of the AV stream file is computed by SPN_xxx = RSPN_xxx-offset_SPN. The value of RSPN_EP_start must appear in ascending order in for-loop of syntax.

[0251] Next, TU_map is explained with reference to drawing 73 . TU_map makes one time-axis based on the arrival timer clock (clock of the arrival time base) of a source packet. The time-axis is called TU_map_time_axis. The zero of TU_map_time_axis is shown by offset_time in TU_map(). TU_map_time_axis is divided into a fixed unit from offset_time. The unit is called time_unit.

[0252] In each time_unit in AV stream, the address on AV stream file of the source packet of the first perfect form is stored in TU_map. These addresses are called RSPN_time_unit_start. It sets on TU_map_time_axis and is k. The time of day when time_unit of eye watch (k>=0) starts is called TU_start_time (k). This value is computed based on a degree type.

TU_start_time (k) = offset_time+k*time_unit_sizeTU_start_time (k) has the precision of 45kHz.

[0253] Drawing 75 is drawing showing the syntax of TU_map. The field of the 32-bit length of offset_time gives the offset time to TU_map_time_axis for explaining the syntax of TU_map shown in drawing 75 . This value shows the offset time of day to time_unit of the beginning in Clip. offset_time is magnitude which makes a unit the 45kHz clock drawn from the arrival timer clock of 27MHz precision. offset_time must be set to zero when AV stream is recorded as new Clip.

[0254] 32 bit fields of time_unit_size give the magnitude of time_unit, and it is

magnitude which makes a unit the 45kHz clock drawn from the arrival timer clock of 27MHz precision. time_unit_size is good to make it 1 or less (time_unit_size<=45000) second. 32 bit fields of number_of_time_unit_entries show the number of entries of time_unit currently stored in TU_map().

[0255] 32 bit fields of RSPN_time_unit_start show the relative address of the location which each time_unit starts in AV stream. RSPN_time_unit_start is magnitude which makes a source packet number a unit, and counts the value of offset_SPN defined in ClipInfo() from the source packet of the beginning of an AV stream file as initial value. The absolute address in the inside of the AV stream file is computed by SPN_xxx = RSPN_xxx-offset_SPN. The value of RSPN_time_unit_start must appear in ascending order in for-loop of syntax. (k+1) When anything does not have a source packet into time_unit of eye watch, RSPN_time_unit_start of eye watch (k+1) must be equal to k-th RSPN_time_unit_start.

[0256] ClipMark in the syntax of zzzzz.clip shown in drawing 45 is explained. ClipMark is the mark information about a clip and is stored into ClipMark. This mark is not set by the recorder (record regenerative apparatus 1), and is not set by the user.

[0257] Drawing 75 is drawing showing the syntax of ClipMark. They are four character alphabetic characters in which version_number shows the version number of this ClipMark() for explaining the syntax of ClipMark shown in drawing 75 . version_number must be encoded with "0045" according to ISO 646.

[0258] length is a 32-bit unsigned integer which shows the byte count of ClipMark() from immediately after this length field to the last of ClipMark(). number_of_Clip_marks, the 16-bit unsigned integer which shows the number of the mark currently stored in ClipMark. number_of_Clip_marks You may be 0. mark_type is the 8-bit field which shows the type of a mark, and is encoded according to the table shown in drawing 76 .

[0259] mark_time_stamp is 32 bit fields and stores the time stump in which the point with which the mark was specified is shown. The semantics of mark_time_stamp changes with CPI_type in PlayList(), as shown in drawing 77 .

[0260] When, as for STC_sequence_id, CPI_type in CPI() shows EP_map type, this 8-bit field shows STC_sequence_id of the STC continuation section on which the mark is put. When CPI_type in CPI() shows TU_map type, this 8-bit field has no semantics, but is set to zero. The 8-bit field of character_set shows the coding approach of the character alphabetic character encoded by the mark_name field. The coding approach corresponds to the value shown in drawing 19 .

[0261] Eight bit fields of name_length show the cutting tool length of the mark name shown in the Mark_name field. The field of mark_name shows the name of a mark. The byte count of the left in this field to a name_length number is an effective character alphabetic character, and it shows the name of a mark. In the mark_name field, as for the value after these effective characters alphabetic character, what kind of value may be in close.

[0262] The field of ref_thumbnail_index shows the information on the thumbnail image added to a mark. In the case of the value whose ref_thumbnail_index field is not 0xFFFF, the thumbnail image is added to the mark and the thumbnail image is stored in the mark.thmb file. The image is referred to using the value of ref_thumbnail_index in a mark.thmb file. the ref_thumbnail_index field -- 0xFFFF it is -- a case -- the mark -- a thumbnail image -- adding -- having -- **** .

[0263] Since MakersPrivateData was already explained with reference to drawing 22 , the explanation is omitted.

[0264] Next, a thumbnail information (Thumbnail Information) is explained. A thumbnail image is stored in a menu.thmb file or a mark.thmb file. These files are the same syntax structures and have only one Thumbnail(). A menu.thmb file stores a menu thumbnail image, i.e., the image representing Volume, and the image representing each PlayList. All menu thumbnails are stored only in one menu.thmb file.

[0265] A mark.thmb file stores the picture showing a mark thumbnail image, i.e., a marking point. All the mark thumbnails to all PlayList(s) and Clip(s) are stored only in one mark.thmb file. Since a thumbnail is added frequently and deleted, add operation and actuation of partial deletion must be able to be easily performed at a high speed. Thumbnail() has the block structure for this

reason. The data of an image are divided into some parts and each part is stored in one tn_block. One image data is stored in tn_block which ******(ed). tn_block which is not used may exist in the train of tn_block. The cutting tool length of one thumbnail image is adjustable.

[0266] Drawing 78 is drawing showing the syntax of menu.thmb and mark.thmb, and drawing 79 is drawing showing the syntax of Thumbnail in the syntax of menu.thmb shown in drawing 78 , and mark.thmb. They are four character alphabetic characters in which version_number shows the version number of this Thumbnail() for explaining the syntax of Thumbnail shown in drawing 79 . version_number must be encoded with "0045" according to ISO 646.

[0267] length is a 32-bit unsigned integer which shows the byte count of MakersPrivateData() from immediately after this length field to the last of Thumbnail(). tn_blocks_start_address is a 32-bit unsigned integer which shows the head byte address of the first tn_block by making the relative byte count from the cutting tool of the head of Thumbnail() into a unit. A relative byte count is counted from zero. number_of_thumbnails is a 16-bit unsigned integer which gives the number of entries of the thumbnail image contained in Thumbnail().

[0268] tn_block_size is a 16-bit unsigned integer which gives the magnitude of one tn_block by making 1024 bytes into a unit. For example, if it becomes tn_block_size=1, it shows that the magnitude of one tn_block is 1024 bytes. number_of_tn_blocks is a 116-bit unsigned integer showing the number of entries of tn_block in this Thumbnail(). thumbnail_index is a 16-bit unsigned integer showing the index number of the thumbnail image expressed with the thumbnail information on the "for" loop batch which begins from this thumbnail_index field. thumbnail_index Don't carry out and don't use the value of 0xFFFF. thumbnail_index Refer to for ref_thumbnail_index in UIAppInfoVolume(), UIAppInfoPlayList(), PlayListMark(), and ClipMark().

[0269] thumbnail_picture_format is a 8-bit unsigned integer showing a picture format of a thumbnail image, and takes a value as shown in drawing 80 . DCF and PNG in a table are allowed only within "menu.thmb." A mark thumbnail must take value"0x00" (MPEG-2 Video I-picture).

[0270] picture_data_size is a 32-bit unsigned integer which shows the cutting tool length of a thumbnail image per cutting tool. start_tn_block_number is a 16-bit unsigned integer showing the tn_block number of tn_block from which the data of a thumbnail image begin. The head of thumbnail image data must be in agreement with the head of tb_block. A tn_block number begins from 0 and is related to the value of the variable k in the "for" loop of tn_block.
[0271] x_picture_length is a 16-bit unsigned integer showing the horizontal number of pixels of the frame picture frame of a thumbnail image. y_picture_length is a 16-bit unsigned integer showing the number of pixels of the perpendicular direction of the frame picture frame of a thumbnail image. tn_block, It is the field in which a thumbnail image is stored. All tn_block in Thumbnail() is the same sizes (fixed length), and the magnitude is defined by tn_block_size.
[0272] Drawing 81 is drawing which meant typically how thumbnail image data would be stored in tn_block. Like drawing 81 , each thumbnail image data begins from the head of tn_block, and, in the case of the magnitude exceeding 1 tn_block, it is stored using continuous following tn_block. By doing in this way, the picture data which is variable length becomes possible [managing as fixed-length data], and can respond now by simple processing to edit called deletion.
[0273] Next, AV stream file is explained. AV stream file is stored in an "M2TS" directory (drawing 14). There are two types of AV stream files, and they are a Clip AV stream and a Bridge-Clip AV stream file. It must be the structure of a DVR MPEG-2 transport stream file where both AV streams are defined henceforth [this].
[0274] First, DVR MPEG-2 A transport stream is explained. The structure of a DVR MPEG-2 transport stream is shown in drawing 82 . AV stream file has the structure of a DVR MPEG 2 transport stream. A DVR MPEG 2 transport stream consists of Aligned unit of an integer individual. the magnitude of Alignedunit -- 6144 Cutting tool (2048*3 cutting tool) it is . Aligned unit begins from the 1st byte of a source packet. A source packet is 192-byte length. One source packet consists of TP_extra_header and transport Paquette. TP_extra_header is 4-byte length and transport Paquette is 188-byte length.

[0275] One Aligned unit consists of 32 source packets. Aligned unit of the last in a DVR MPEG 2 transport stream also consists of 32 source packets. Therefore, termination of the DVR MPEG 2 transport stream is carried out on the boundary of Aligned unit. When the number of transport Paquette of the input transport stream recorded on a disk is not a multiple of 32, a source packet with Nur Paquette (transport Paquette of PID=0x1FFF) must be used for the last Aligned unit. A file system must not add excessive information to a DVR MPEG 2 transport stream.

[0276] The recorder model of a DVR MPEG-2 transport stream is shown in drawing 83 . The recorder shown in drawing 83 is a model on the concept for specifying a recording process. A DVR MPEG-2 transport stream follows this model.

[0277] The input timing of an MPEG-2 transport stream is explained. An input MPEG 2 transport stream is a full transport stream or a partialness transport stream. The MPEG 2 transport stream inputted must follow ISO/IEC 13818-1 or ISO/IEC 13818-9. The i-th cutting tool of an MPEG 2 transport stream is simultaneously inputted into T-STD (Transport stream system target decoder specified by ISO/IEC 13818-1), and sow spa KETTAIZA at time-of-day t (i). Rpk is the instant-maximum of transport Paquette's input rate.

[0278] PLL52 generates 27MHz of frequencies of a 27MHz clock. The frequency of a 27MHz clock is locked by the value of PCR (Program Clock Reference) of an MPEG-2 transport stream. arrival time clock counter53 is a binary counter which counts a pulse with a frequency of 27MHz. Arrival_time_clock (i) is the counted value of Arrival time clock counter in time-of-day t (i).

[0279] source packetizer54 adds TP_extra_header to all transport Paquette, and makes a source packet. Arrival_time_stamp expresses the time of day when transport Paquette's 1st byte arrives to both T-STD and sow spa KETTAIZA. Arrival_time_stamp (k) is the sampled value of Arrival_time_clock (k), as shown in a degree type, and k shows transport Paquette's 1st byte here.

arrival_time_stamp (k) = arrival_time_clock(k) % 230[0280] When two time intervals of transport Paquette inputted continuously become 230 / more than

27 million second (about 40 seconds), the two difference of transport Paquette's arrival_time_stamp should be set as it has been 230 / 27 million seconds. It has the recorder, when becoming such.

[0281] smoothing buffer55 carries out smoothing of the bit rate of an input transport stream. Don't overflow a smoothing buffer. Rmax is the output bit rate of the source packet from a smoothing buffer in case a smoothing buffer is not empty. When a smoothing buffer is empty, the output bit rate from a smoothing buffer is zero.

[0282] Next, the parameter of the recorder model of a DVR MPEG-2 transport stream is explained. The value of Rmax is given by TS_recording_rate defined in ClipInfo() corresponding to AV stream file. This value is computed by the degree type.

The value of Rmax = TS_recording_rate * 192/188 TS_recording_rate is magnitude which makes bytes/second a unit.

[0283] When an input transport stream is a SESF transport stream, Rpk must be equal to TS_recording_rate defined in ClipInfo() corresponding to AV stream file. When an input transport stream is not a SESF transport stream, refer to the value defined in a descriptor, for example, maximum_bitrate_descriptor, partial_transport_stream_descriptor, etc. of MPEG-2 transport stream for this value.

[0284] When the input transport stream of smoothing buffer size is a SESF transport stream, the magnitude of a smoothing buffer is zero. When an input transport stream is not a SESF transport stream, refer to the value defined in the descriptor of MPEG-2 transport stream, for example, smoothing_buffer_descriptor, short_smoothing_buffer_descriptor, partial_transport_stream_descriptor, etc. for the magnitude of a smoothing buffer.

[0285] A record machine (recorder) and a playback machine (player) must prepare the buffer of sufficient size. default buffer size -- 1536 bytes it is .

[0286] Next, the player model of a DVR MPEG-2 transport stream is explained. Drawing 84 is drawing showing the player model of a DVR MPEG-2 transport stream. This is a model on the concept for specifying reconstructive processing. A DVR MPEG-2 transport stream follows this

model.

[0287] 27 MHz X-tal61 generates the frequency of 27Mhz. The error range of a 27MHz frequency must be +/-30 ppm (27 million+/-810 Hz). arrival timeclock counter62 is a binary counter which counts a pulse with a frequency of 27MHz. Arrival_time_clock (i) is the counted value of Arrival time clock counter in time-of-day t (i).

[0288] In smoothing buffer64, Rmax is the input bit rate of the source packet to a smoothing buffer in case a smoothing buffer is not full. When a smoothing buffer is full, the input bit rate to a smoothing buffer is zero.

[0289] When arrival_time_stamp of the present source packet is equal to the value which is 30 bits of LSB of arrival_time_clock (i) for explaining the output timing of an MPEG-2 transport stream, transport Paquette, the source packet, is lured from a smoothing buffer. Rpk is the instant-maximum of a transport packet rate. Don't carry out the underflow of the smoothing buffer.

[0290] About the parameter of the player model of a DVR MPEG-2 transport stream, it is the same as that of the parameter of the recorder model of a DVR MPEG-2 transport stream mentioned above.

[0291] Drawing 85 is drawing showing the syntax of Source packet. transport_packet() is MPEG-2 transport Paquette specified by ISO/IEC 13818-1. The syntax of TP_Extra_header in the syntax of Source packet shown in drawing 85 is shown in drawing 86 . It is the integer as which copy_permission_indicator expresses a copy limit of transport Paquette's pay load for explaining the syntax of TP_Extra_header shown in drawing 86 . A copy limit can be set to copy free, no more copy, copy once, or copy prohibited. Drawing 87 shows the value of copy_permission_indicator, and the relation in the mode specified by them.

[0292] copy_permission_indicator is added to all transport Paquette. The value of copy_permission_indicator may be related with the value of EMI in IEEE1394 isochronouspacket header (Encryption Mode Indicator) when recording an input transport stream using an IEEE1394 digital interface. The value of copy_permission_indicator may be related with the value of CCI embedded into transport Paquette, when recording an input transport stream without using an IEEE1394 digital interface. The value of

copy_permission_indicator may be related with the value of CGMS-A of an analog signal when carrying out self encoding of the analog signal input.

[0293] arrival_time_stamp is degree type arrival_time_stamp (k). In = arrival_time_clock(k) % 230, it is an integral value with the value specified by arrival_time_stamp.

[0294] A Clip AV stream must have [defining a Clip AV stream and] the structure of a DVR MPEG-2 transport stream where a definition which was mentioned above is carried out. arrival_time_clock (i) must increase continuously in a Clip AV stream. Even if the break point of system time base (STC base) exists in a Clip AV stream, arrival_time_clock (i) of the Clip AV stream must increase continuously.

[0295] The maximum of the difference of the initiation in a Clip AV stream and arrival_time_clock between termination (i) must be 26 hours. This limit guarantees that PTS (Presentation Time Stamp) of the same value never appears in a Clip AV stream, when the break point of system time base (STC base) does not exist in an MPEG 2 transport stream. MPEG 2 systems specification has specified the wrap around period of PTS as 233 / 90000 second (about 26.5 hours) .

[0296] A Bridge-Clip AV stream must have [defining a Bridge-Clip AV stream and] the structure of a DVR MPEG-2 transport stream where a definition which was mentioned above is carried out. A Bridge-Clip AV stream must contain the break point of one arrival time base. The transport stream before and behind the break point of arrival time base must follow DVR-STD which must follow a limit of coding mentioned later and is mentioned later.

[0297] In the gestalt of this operation, the video between PlayItem(s) in edit and seamless connection of an audio are supported. Making between PlayItem seamless connection guarantees "continuation supply of data", and "seamless decode processing" to a player/recorder. "Continuation supply of data" is being able to guarantee a file system supplying data with a required bit rate so that a decoder's may not be made to cause the underflow of a buffer. The real time nature of data is guaranteed, and data are stored in the block unit which sufficient magnitude followed so that data can be read from a disk.

[0298] "Seamless decode processing" is that a player can display the audio video data recorded on the disk, without making the playback output of a decoder start a pause and a gap.

[0299] AV stream which PlayItem by which seamless connection is made refers to is explained. It can judge whether connection of PlayItem to precede and the present PlayItem is guaranteed to indicate by seamless from the connection_condition field defined in the present PlayItem. The seamless connection between PlayItem(s) has the approach of using Bridge-Clip, and the approach which is not used.

[0300] Drawing 88 shows the relation between PlayItem preceded in the case of using Bridge-Clip, and the present PlayItem. In drawing 88 , the stream data which a player reads give a shadow and are shown. TS1 shown in drawing 88 consists of the stream data which were able to attach the shadow of Clip1 (Clip AV stream), and the stream data which were able to attach the shadow before RSPN_arrival_time_discontinuity of Bridge-Clip.

[0301] The stream data which were able to attach the shadow of Clip1 of TS1 are stream data from the address of a stream required in order to decode the presentation unit corresponding to IN_time (illustrated by IN_time1 in drawing 88) of PlayItem to precede to the source packet referred to by RSPN_exit_from_previous_Clip. The stream data which were able to attach the shadow before RSPN_arrival_time_discontinuity of Bridge-Clip contained in TS1 are stream data from the source packet of the beginning of Bridge-Clip to the source packet in front of the source packet referred to by RSPN_arrival_time_discontinuity.

[0302] Moreover, TS2 in drawing 88 consists of the stream data which were able to attach the shadow of Clip2 (Clip AV stream), and the stream data which were able to attach the shadow after RSPN_arrival_time_discontinuity of Bridge-Clip. The stream data which were able to attach the shadow after RSPN_arrival_time_discontinuity of Bridge-Clip contained in TS2 are stream data from the source packet referred to by RSPN_arrival_time_discontinuity to the source packet of the last of Bridge-Clip. The stream data which were able to attach the shadow of Clip2 of TS2 are stream data to the address of a stream required in order to decode the presentation unit corresponding to

OUT_time (illustrated by OUT_time2 in drawing 88) of the present PlayItem from the source packet referred to by RSPN_enter_to_current_Clip.

[0303] Drawing 89 shows the relation between PlayItem preceded when not using Bridge-Clip, and the present PlayItem. In this case, the stream data which a player reads give a shadow and are shown. TS1 in drawing 89 consists of the stream data which were able to attach the shadow of Clip1 (Clip AV stream). The stream data which were able to attach the shadow of Clip1 of TS1 begin from the address of a stream required in order to decode the presentation unit corresponding to IN_time (illustrated by IN_time1 in drawing 89) of PlayItem to precede, and are data to the source packet of the last of Clip1. Moreover, TS2 in drawing 89 consists of the stream data which were able to attach the shadow of Clip2 (Clip AV stream).

[0304] The stream data which were able to attach the shadow of Clip2 of TS2 are stream data to the address of a stream required in order to begin from the source packet of the beginning of Clip2 and to decode the presentation unit corresponding to OUT_time (illustrated by OUT_time2 in drawing 89) of the present PlayItem.

[0305] In drawing 88 and drawing 89 , TS1 and T2 are the streams which the source packet followed. Next, a stream convention of TS1 and TS2 and the connection conditions between them are considered. First, the coding limit for seamless connection is considered. As a limit of the coding structure of a transport stream, the number of the programs included in TS1 and TS2 must be 1 first. The number of the video streams contained in TS1 and TS2 must be 1. The number of the audio streams contained in TS1 and TS2 must be two or less. The number of the audio streams contained in TS1 and TS2 must be equal. In TS1 and/or TS2, the elementary streams or private streams other than the above may be contained.

[0306] A limit of a video bit stream is explained. Drawing 90 is drawing showing the example of the seamless connection shown by the display order of a picture. In order to be able to display a video stream seamlessly in a node, the unnecessary picture displayed before IN_time2 (IN_time of Clip2) the OUT_time1 (OUT_time of Clip1) back must be removed by the process which re-encodes the partial stream of Clip near a node.
 [0307] The example

which makes seamless connection using BridgeSequence when shown in drawing 90 is shown in drawing 91 . The video stream of Bridge-Clip before RSPN_arrival_time_discontinuity consists of the coding video stream to the picture corresponding to OUT_time1 of Clip1 of drawing 90 . And it connects with the video stream of Clip1 to precede, and the video stream is re-encoded so that it may become the elementary stream which followed MPEG 2 specification by one continuation.

[0308] Similarly, the video stream of Bridge-Clip after RSPN_arrival_time_discontinuity consists of the coding video stream after the picture corresponding to IN_time2 of Clip2 of drawing 90 . And decoding initiation can be carried out correctly and it connects with the video stream of Clip2 following this, and the video stream is re-encoded so that it may become the elementary stream which followed MPEG 2 specification by one continuation. In order to make Bridge-Clip, generally, the picture of several sheets must be re-encoded and the other picture can be copied from Clip of an original copy.

[0309] The example which makes seamless connection without using BridgeSequence in the case of the example shown in drawing 90 is shown in drawing 92 . The video stream of Clip1 consists of the coding video stream to the picture corresponding to OUT_time1 of drawing 90 , and it is re-encoded so that it may become the elementary stream which followed MPEG 2 specification by one continuation. Similarly, the video stream of Clip2 consists of the coding video stream after the picture corresponding to IN_time2 of Clip2 of drawing 90 , and it is re-encoded so that it may become the elementary stream which followed MPEG 2 specification by one continuation.

[0310] The frame rate of the video stream of TS1 and TS2 must be equal to explaining a coding limit of a video stream first. Termination of the video stream of TS1 must be carried out by sequence_end_code. The video stream of TS2 must be started by Sequence Header, GOP Header, and I-picture. the video stream of TS2 -- closed one -- it must start by GOP.

[0311] The video presentation unit (a frame or field) defined in a bit stream must be continuation on both sides of a node. There must not be no gap of a frame or the field in a node. In a node, the field sequence of a top ? bottom

product must be continuation. In encoding which uses 3-2 PURUDAUN, it is "top_field_first". It reaches. In order to rewrite a "repeat_first_field" flag or to prevent generating of a field gap, you may make it re-encode locally.

[0312] The sampling frequency of the audio of TS1 and TS2 must be the same as explaining a coding limit of an audio bit stream. The coding approach (example . MPEG1 layer 2, AC-3, SESF LPCM, AAC) of the audio of TS1 and TS2 must be the same.

[0313] Next, the audio frame of the last of the audio stream of TS1 must contain the audio sample with display time of day equal at the time of display termination of the display picture of the last of TS1 in explaining a coding limit of an MPEG-2 transport stream. The audio frame of the beginning of the audio stream of TS2 must contain the audio sample with display time of day equal at the time of display initiation of the display picture of the beginning of TS2.

[0314] In a node, a gap must not be in the sequence of an audio presentation unit. As shown in drawing 93 , there may be overlap defined by the die length of the audio presentation unit of under 2 audio frame section. The first Paquette who transmits the elementary stream of TS2 must be a video packet. The transport stream in a node must follow DVR-STD mentioned later.

[0315] TS1 and TS2 must not contain the break point of arrival time base in explaining a limit of Clip and Bridge-Clip in each.

[0316] The following limits are applied only when using Bridge-Clip. Only in the node of the source packet of the last of TS1, and the source packet of the beginning of TS2, a Bridge-ClipAV stream has the break point of only one arrival time base. RSPN_arrival_time_discontinuity defined in ClipInfo() must show the address of the break point, and it must show the address which refers to the source packet of the beginning of TS2.

[0317] Any source packet in Clip1 is sufficient as the source packet referred to by RSPN_exit_from_previous_Clip defined in BridgeSequenceInfo(). It does not need to be the boundary of Aligned unit. Any source packet in Clip2 is sufficient as the source packet referred to by RSPN_enter_to_current_Clip defined in BridgeSequenceInfo(). It does not need to be the boundary of Aligned unit.

[0318] OUT_time (OUT_time1 shown in drawing 88 and drawing 89) of PlayItem preceded for explaining a limit of PlayItem must show the display end time of the video presentation unit of the last of TS1. IN_time (IN_time2 shown in F drawing 88 and drawing 89) of the present PlayItem must show the display start time of the video presentation unit of the beginning of TS2.
[0319] Seamless connection must be made by explaining a limit of the data allocation in the case of using Bridge-Clip with reference to drawing 94 so that continuation supply of data may be guaranteed by the file system. This must be performed by arranging the Bridge-Clip AV stream connected to Clip1 (Clip AV stream file) and Clip2 (Clip AV stream file) so that a data allocation convention may be fulfilled.
[0320] RSPN_exit_from_previous_Clip must be chosen as the stream part of Clip1 (Clip AV stream file) before RSPN_exit_from_previous_Clip is arranged to the continuation field more than half fragmentation. The data length of a Bridge-Clip AV stream must be chosen so that it may be arranged to the continuation field more than half fragmentation. RSPN_enter_to_current_Clip must be chosen as the stream part of Clip2 (Clip AV stream file) after RSPN_enter_to_current_Clip is arranged to the continuation field more than half fragmentation.
[0321] Seamless connection must be made by explaining a limit of the data allocation in the case of making seamless connection without using Bridge-Clip with reference to drawing 95 so that continuation supply of data may be guaranteed by the file system. This must be performed by arranging the part of the last of Clip1 (Clip AV stream file), and the part of the beginning of Clip2 (Clip AV stream file) so that a data allocation convention may be fulfilled.
[0322] The stream part of the last of Clip1 (Clip AV stream file) must be arranged to the continuation field more than half fragmentation. The stream part of the beginning of Clip2 (Clip AV stream file) must be arranged to the continuation field more than half fragmentation.
[0323] Next, DVR-STD is explained. DVR-STD is a conceptual model for modeling generation of a DVR MPEG 2 transport stream, and decoding in the case of verification. Moreover, DVR-STD is also a conceptual model for modeling generation of AV stream referred to by two PlayItem(s) which were

mentioned above, and by which seamless connection was made, and decoding in the case of verification.

[0324] A DVR-STD model is shown in drawing 96 . The DVR MPEG-2 transport stream player model is contained in the model shown in drawing 96 as a component. The notation approach of of n, TBn, MBn, EBn, TBsys, Bsys, Rxn, Rbxn, Rxsys, Dn, Dsys, and On and Pn (k) is the same as what is defined as T-STD of ISO/IEC 13818-1. That is, it is as follows. n is the index number of an elementary stream. TBn is the transport buffer of the elementary stream n, and is **.

[0325] MBn is the multiplex buffer of the elementary stream n. It exists only about a video stream. EBn is the elementary stream buffer of the elementary stream n. It exists only about a video stream. TBsys is an input buffer for the system information of the program under decode. Bsys is a main buffer in the system target decoder for the system information of the program under decode. Rxn is a transmission rate by which data are removed from TBn. Rbxn is a transmission rate by which a PES Paquette pay load is removed from MBn. It exists only about a video stream.

[0326] Rxsys is a transmission rate by which data are removed from TBsys. Dn is the decoder of the elementary stream n. Dsys is a decoder about the system information of the program under decode. On is re-ordering buffer of the video stream n. Pn (k) is the k-th presentation unit of the elementary stream n.

[0327] The decoding process of DVR-STD is explained. While reproducing the single DVR MPEG-2 transport stream, the timing which inputs transport Paquette into the buffer of TB1, TBn, or TBsys is determined by arrival_time_stamp of a source packet. The convention of buffering actuation of TB1, MB1, EB1, TBn, Bn, TBsys, and Bsys is the same as T-STD specified to ISO/IEC 13818-1. A convention of decode actuation and a display action is the same as T-STD specified to ISO/IEC 13818-1.

[0328] A decoding process while reproducing PlayItem by which seamless connection was made is explained. Here, playback of two AV streams referred to by PlayItem by which seamless connection was made will be explained, and future explanation explains the playback of TS (for example,

shown in drawing 88)1, and TS2 mentioned above. TS1 is a stream to precede and TS2 is a current stream.

[0329] Drawing 97 shows the timing chart of the input of transport Paquette when moving from a certain AV stream (TS1) to the following AV stream (TS2) seamlessly connected to it, decode, and a display. While moving from predetermined AV stream (TS1) to the following AV stream (TS2) seamlessly connected to it, the time-axis (drawing 97 is shown by ATC2) of the arrival time base of TS2 is not the same as the time-axis (drawing 97 is shown by ATC1) of the arrival time base of TS1.

[0330] Moreover, the time-axis (drawing 97 is shown by STC2) of the system time base of TS2 is not the same as the time-axis (drawing 97 is shown by STC1) of the system time base of TS1. It is required that the display of video should continue seamlessly. Overlap may be shown in the display time of the presentation unit of an audio.

[0331] DVR-STD Input timing is explained. It is TB1 and TBn of DVR-STD until the time amount by time of day T1, i.e., the video packet of the last of TS1, carries out input termination at TB1 of DVR-STD. Or the input timing to the buffer of TBsys is determined by arrival_time_stamp of the source packet of TS1.

[0332] Remaining Paquette of TS1 must be inputted into the buffer of TBn of DVR-STD, or TBsys with the bit rate of TS_recording_rate (TS1). Here, TS_recording_rate (TS1) is the value of TS_recording_rate defined in ClipInfo() corresponding to Clip1. The time of day which the cutting tool of the last of TS1 inputs into a buffer is time of day T2. Therefore, arrival_time_stamp of a source packet is disregarded in the section from time of day T1 to T2.

[0333] If N1 is made into the byte count of transport Paquette of TS1 following the video packet of the last of TS1, time of day T1 thru/or the time amount DT 1 to T2 will be time amount required in order that 1 byte of N may carry out input termination with the bit rate of TS_recording_rate (TS1), and will be computed by the degree type.

Both the values of RXn and RXsys change to the value of TS_recording_rate (TS1) before the DT1=T2-T1=N1 / TS_recording_rate (TS1) time of day T1

thru/or T2. The buffering actuation of those other than this rule is the same as T-STD.

[0334] arrival time clock counter is reset by the value of arrival_time_stamp of the source packet of the beginning of TS2 in the time of day of T2. TB1 of DVR-STD, and TBn Or the input timing to the buffer of TBsys is determined by arrival_time_stamp of the source packet of TS2. RXn and RXsys both change to the value defined in T-STD.

[0335] In addition to the amount of buffers defined by T-STD, an audio decoder and a system decoder need the additional amount of buffers (amount of data for about 1 second) to explain additional audio buffering and system data buffering so that the input data of the section from time of day T1 to T2 can be processed.

[0336] The display of a video presentation unit must let a node pass to explain the presentation timing of video, and it must be continuation without a gap. Here, STC1 considers as the time-axis (in drawing 97 , illustrated with STC1) of the system time base of TS1, and STC2 is the time-axis (in drawing 97 , illustrated with STC2.) of the system time base of TS2. In accuracy, PCR of the beginning of TS2 starts STC2 from the time of day inputted into T-STD. It carries out.

[0337] The offset between STC1 and STC2 is determined as follows. end is PTS on PTS1STC1 corresponding to the video presentation unit of the last of TS1, PTS2start is PTS on STC2 corresponding to the video presentation unit of the beginning of TS2, and if Tpp considers as the display period of the video presentation unit of the last of TS1, offset STC_delta between two system time base will be computed by the degree type.

STC_delta = PTS1end+Tpp-PTS2start [0338] Explaining the timing of the presentation of an audio may have the overlap of the display timing of an audio presentation unit in a node, and they are 0 thru/or under 2 audio frame (see "audio overlap" currently illustrated by drawing 97). Which audio sample being chosen and carrying out resynchronization of the display of an audio presentation unit to the time base where it was amended after the node are set up by the player side.

[0339] In time of day T5, the audio presentation unit of the last of TS1 is

displayed for explaining about the system time clock of DVR-STD. The system time clock may overlap among T5 from time of day T2. In this section, DVR-STD changes a system time clock between the value (STC1) of old time base, and the value (STC2) of new time base. The value of STC2 is computed by the degree type.
STC2=STC1-STC_delta [0340] The continuity of buffering is explained. STC11 video_end is the value of STC on the system time base STC 1 in case the cutting tool of the last of the video packet of the last of TS1 arrives to TB1 of DVR-STD. STC22 video_start is the value of STC on the system time base STC 2 in case the cutting tool of the beginning of the video packet of the beginning of TS2 arrives to TB1 of DVR-STD. STC21 video_end is STC11 video_end. It is the value which converted the value into the value on the system time base STC 2. STC21 video_end is computed by the degree type.
STC21 video_end = STC11 video_end -STC_delta [0341] In order to follow DVR-STD, it is required that the following two conditions should be fulfilled. First, the arrival timing to TB1 of the video packet of the beginning of TS2 must fill the inequality shown below. And the inequality shown below must be filled.
STC22 video_start > STC21 video_end+ deltaT1 -- this inequality is filled -- as -- Clip1 -- and -- or the partial stream of Clip2 -- re-encoding -- and -- or when it is necessary to re-multiplex, it is carried out if needed [that].
[0342] next, the input of the video packet from TS2 which continues at the input of the video packet from TS1, and it on the time-axis of the system time base which converted STC1 and STC2 on the same time-axis -- a video buffer -- overflow -- and don't carry out an underflow.
[0343] The content of the data currently recorded on the record medium, playback information, etc. can be managed appropriately, and it has them, and a user can check the content of the data currently recorded on the record medium appropriately at the time of playback, or it can make it possible to reproduce desired data simple by being based on such syntax, DS, and a regulation.
[0344] In addition, although the gestalt of this operation makes an MPEG 2 transport stream an example and explains it as a multiplexing stream, it can

be applied also about the DSS transport stream currently used with DirecTV service (trademark) of not only this but an MPEG 2 program stream, or the U.S.

[0345] Next, drawing 98 shows another example of a PlayList file. The big difference in the syntax of drawing 98 and drawing 23 is a location in which UIAppInfoPlayList() is stored. In the example of drawing 98 , since UIAppInfoPlayList() is taken out outside out of PlayList(), the future information escape of UIAppInfoPlayList() can carry out comparatively easily.

[0346] version_number is four figures which show the version number of this thumbnail header information file.

[0347] PlayList_start_address shows the start address of PlayList() by making the relative byte count from the cutting tool of the head of a PlayList file into a unit. A relative byte count is counted from zero.

[0348] PlayListMark_start_address shows the start address of PlayListMark() by making the relative byte count from the cutting tool of the head of a PlayList file into a unit. A relative byte count is counted from zero.

[0349] MakersPrivateData_start_address shows the start address of MakersPrivateData() by making the relative byte count from the cutting tool of the head of a PlayList file into a unit. A relative byte count is counted from zero.

[0350] Drawing 99 shows the syntax of UIAppInfoPlayList in the PlayList file of drawing 98 . PlayList_service_type shows the type of a PlayList file. The example is shown in drawing 26 . Moreover, PlayList_service_type may give the same semantics as the service type which the program of digital TV broadcast shows. For example, in digital BS broadcast of Japan, a service type has three kinds, television service, voice service, and data broadcast service. The value representing the service type of the program which the Clip AV stream which PlayList uses includes is set to PlayList_service_type.

[0351] PlayList_character_set shows the coding approach of the character alphabetic character encoded by channel_name, PlayList_name, and the PlayList_detail field. Moreover, this shows the coding approach of the character alphabetic character encoded by the mark_name field in PlayListMark.

[0352] channel_number shows the broadcast channel number or service number chosen by the user, when the PlayList is recorded. When the combined harvester and thresher of two or more PlayList(s) is carried out to one PlayList, this field shows the central value of that PlayList. When this field is set to 0xFFFF, this field has no semantics.

[0353] channel_name_length shows the cutting tool length of the channel name shown in the channel_name field. This field is 20 or less value.

[0354] channel_name shows the broadcast channel chosen by the user or the identifier of service, when the PlayList is recorded. A number of byte counts shown by channel_name_length from the left in this field are effective character alphabetic characters, and said identifier is shown. As for the remaining cutting tools who follow these effective characters alphabetic character in this field, what kind of value may be set. When the combined harvester and thresher of two or more PlayList(s) is carried out to one PlayList, this field shows the identifier representing that PlayList.

[0355] PlayList_name_length shows the cutting tool length of the PlayList name shown in the PlayList_name field.

[0356] PlayList_name shows the identifier of PlayList. A number of byte counts shown by PlayList_name_length from the left in this field are effective character alphabetic characters, and said identifier is shown. As for the remaining cutting tools who follow these effective characters alphabetic character in this field, what kind of value may be set.

[0357] PlayList_detail_length shows the cutting tool length of the detailed information of PlayList shown in the PlayList_datail field. This field is 1200 or less value.

[0358] PlayList_detail shows the text explaining the detailed information of PlayList. A number of byte counts shown by PlayList_detail_length from the left in this field are effective character alphabetic characters, and said text is shown. As for the remaining cutting tools who follow these effective characters alphabetic character in this field, what kind of value may be set.

[0359] The semantics of the syntax fields other than this is the same as the field of a same name shown in drawing 27 .

[0360] Drawing 100 shows the syntax of PlayList() in the PlayList file of

drawing 98 . It is only different in that UIAppInfoPlayList() was lost compared with the example of drawing 25 , and fundamentally the same except this.

[0361] Drawing 101 shows example of another of the syntax of SubPlayItem. The point that STC_sequence_id was added compared with the example of drawing 40 is a big difference.

[0362] STC_sequence_id shows STC_sequence_id of STC which SubPath_IN_time and SubPath_OUT_time for specifying the playback section on AV stream file corresponding to Clip_Information_file_name refer to. SubPath_IN_time and SubPath_OUT_time show the time amount on the same STC continuation section specified by STC_sequence_id.

[0363] By adding STC_sequence_id to SubPlayItem, AV stream file which SubPlayItem refers to comes to be allowed to have an STC break point.

[0364] The semantics of the syntax fields other than this is the same as the field of a same name shown in drawing 40 .

[0365] Drawing 102 shows the flow chart explaining the creation approach of Real PlayList. It explains referring to the block diagram of the record regenerative apparatus of drawing 1 .

[0366] At step S11, a control section 23 records a Clip AV stream.

[0367] At step S12, it investigates whether a control section 23 can create EP_map of a Clip AV stream. At step S12, in Yes, it progresses to step S13, and it creates EP_map. At step S12, in No, it progresses to step S14, and it creates TU_map.

[0368] Then, a control section 23 sets CPI_type of PlayList at step S15.

[0369] At step S16, a control section 23 creates PlayList() which consists of PlayItem which covers all the refreshable range of Above Clip. When CPI_type is an EP_map type, a hour entry is set with the PTS base, an STC break point is in Clip and PlayList() consists [this] of two or more PlayItem(s), connection_condition between PlayItem(s) is also determined. When CPI_type is a TU_map type, a hour entry is set in arrival time base.

[0370] At step S17, a control section 23 creates UIAppInfoPlayList().

[0371] At step S18, a control section 23 creates PlayListMark.

[0372] At step S19, a control section 23 creates MakersPrivateData.

[0373] At step S20, a control section 23 records a Real PlayList file.

[0374] Thus, whenever it records a Clip AV stream newly, one Real PlayList file is made.

[0375] Drawing 103 is a flow chart explaining the creation approach of Virtual PlayList.

[0376] At step S31, it lets a user interface pass and one Real PlayList currently recorded on the disk is specified. And out of the playback range of the Real PlayList, it lets a user interface pass and the playback section shown with IN point and an OUT point is specified. When CPI_type is an EP_map type, the playback section is set with the PTS base, and when CPI_type is a TU_map type, the playback section is set in arrival time base.

[0377] It investigates whether at step S32, all assignment actuation of the playback range by the user ended the control section 23. When a user chooses the section reproduced after the playback section which carried out [above-mentioned] directions, it returns to step S31. When all assignment actuation of the playback range by the user is completed at step S32, it progresses to step S33.

[0378] At step S33, a user determines the connection condition (connection_condition) during the two playback sections reproduced continuously through a user interface, or a control section 23 is determined.

[0379] At step S34, when CPI_type is an EP_map type, it lets a user interface pass and a user specifies subpass (audio for postrecording) information. When a user does not create subpass, this step does not exist.

[0380] At step S35, a control section 23 creates PlayList() based on the playback range information specified by a user, and connection_condition.

[0381] At step S36, a control section 23 creates UIAppInfoPlayList().

[0382] At step S37, a control section 23 creates PlayListMark.

[0383] At step S38, a control section 23 creates MakersPrivateData.

[0384] At step S39, a control section 23 records a Virtual PlayList file.

[0385] Thus, one Virtual PlayList file is made for every thing which chose the playback section which a user wants to see from the playback range of Real PlayList currently recorded on the disk, and carried out grouping of the playback section.

[0386] Drawing 104 is a flow chart explaining the playback approach of

PlayList.

[0387] A control section 23 acquires the information on Info.dvr, Clip Information file, PlayList file, and a thumbnail file, creates the GUI screen in which the list of PlayList currently recorded on the disk is shown, lets a user interface pass, and expresses it to GUI as step S51.

[0388] At step S52, a control section 23 shows a GUI screen the information explaining PlayList based on UIAppInfoPlayList() of each PlayList.

[0389] At step S53, it lets a user interface pass and a user directs playback of one PlayList from on a GUI screen.

[0390] At step S54, a control section 23 acquires the source packet number which has the nearest entry point in front in time than IN_time from STC-sequenc-id of the present PlayItem, and PTS of IN_time, when CPI_type is an EP_map type. Or a control section 23 acquires the source packet number which the nearest time unit in front starts in time than IN_time to IN_time of the present PlayItem, when CPI_type is a TU_map type.

[0391] At step S55, a control section 23 reads the data of AV stream from the source packet number obtained at the above-mentioned step, and supplies them to the AV decoder 27.

[0392] At step S56, when there is front PlayItem in time [current PlayItem], a control section 23 performs connection processing of the display with front PlayItem and current PlayItem according to connection_condition.

[0393] At step S57, a control section 23 directs that the AV decoder 27 starts a display from the picture of PTS of IN_time, when CPI_type is an EP_map type. Or a control section 23 directs that the AV decoder 27 starts a display from the picture of the stream after IN_time, when CPI_type is an EP_map type.

[0394] At step S58, a control section 23 directs to continue decoding of AV stream to the AV decoder 27.

[0395] At step S59, when CPI_type is an EP_map type, as for a control section 23, the image of the present display investigates whether it is the image of PTS of OUT_time. Or a control section 23 investigates whether the stream decoded now passed over OUT_time, when CPI_type is a TU_map type.

[0396] In No, it progresses to step S60 at step S59. An image current at step S60 is displayed, and it returns to step S58. In Yes, it progresses to step S61.
[0397] At step S61, as for a control section 23, current PlayItem investigates in PlayList whether it is the last PlayItem. In No, it returns to step S54. In Yes, playback of PlayList is ended.
[0398] Drawing 105 is a flow chart explaining the playback approach of the Sub pass of PlayList. The playback approach of the subpass of PlayList of drawing 105 is used only when CPI_type of PlayList is EP_map. Processing of this flow chart is simultaneously performed with the processing after step S54 in playback of PlayList of drawing 104 . Moreover, the AV decoder 27 is premised on decoding of two audio streams being simultaneously possible.
[0399] At step S71, a control section 23 acquires the information on SubPlayItem.
[0400] At step S72, a control section 23 acquires the source packet number which has the nearest entry point in front in time than SubPath_IN_time.
[0401] At step S73, a control section 23 reads the data of AV stream of subpass from a source packet number with the above-mentioned entry point, and supplies them to the AV decoder 27.
[0402] At step S74, a control section 23 directs that the audio of subpass starts a display to the AV decoder 27, if playback of Main pass becomes the picture shown by sync_PlayItem_id and sync_start_PTS_of_PlayItem.
[0403] The AV decoder 27 continues decoding of AV stream of subpass at step S75.
[0404] PTS of the subpass in which a control section 23 indicates by current investigates whether it is SubPath_OUT_time at step S76. In No, it progresses to step S77. The display of subpass is continued at step S77, and it returns to step S75.
[0405] In SubPath_OUT_time, PTS of the subpass which indicates by current at step S76 ends the display of subpass.
[0406] Playback of the main path of one PlayList file in which carried out as shown in Fig. 104 and 105 , and playback directions were done by the user, and subpass is performed.
[0407] Drawing 106 shows the flow chart explaining the creation approach of

PlayListMark. It explains referring to the block diagram of the record regenerative apparatus of drawing 1 .

[0408] A control section 23 acquires the information on Info.dvr, Clip Information file, PlayList file, and Thumbnail file, creates the GUI screen in which the list of PlayList currently recorded on the disk is shown, lets a user interface pass, and expresses it to GUI as step S91.

[0409] At step S92, it lets a user interface pass and a user directs playback of one PlayList to a control section 23.

[0410] A control section 23 makes the playback of PlayList by which directions were carried out [above-mentioned] start at step S93 (refer to the drawing 104).

[0411] At step S94, it lets a user interface pass and the set of a mark is directed to a control section 23 at the place whose user is a favorite scene.

[0412] At step S95, a control section 23 acquires PTS of a mark, and PlayItem_id of PlayItem to which it belongs, when CPI_type is EP_map. Or a control section 23 is [0413] which acquires the arrival time of a marking point when CPI_type is TU_map. At step S96, a control section 23 stores the information on a mark in PlayListMark().

[0414] At step S97, a control section 23 records a PlayList file on a record medium 100.

[0415] Drawing 107 is a flow chart explaining the search playback approach which used PlayListMark. It explains referring to the block diagram of the record regenerative apparatus of drawing 1 .

[0416] A control section 23 acquires the information on Info.dvr, Clip Information file, PlayList file, and Thumbnail file, creates the GUI screen in which the list of PlayList currently recorded on the disk (record medium 100) is shown, lets a user interface pass, and expresses it to GUI as step S111.

[0417] At step S112, a control section 23 lets a user interface pass, and a user directs playback of one PlayList.

[0418] A control section 23 expresses a user interface to GUI as step S113 through the list of thumbnails generated from the picture referred to by PlayListMark.

[0419] At step S114, it lets a user interface pass and a user specifies the

marking point of a playback start point as a control section 23.

[0420] As for a control section 23, as for CPI_type, in an EP_map type case, at step S115, PTS of a mark and it acquire PlayItem_id which belongs. Or in a control section 23, in a TU_map type case, CPI_type acquires ATS (Arrival Time Stamp) of a mark.

[0421] As for a control section 23, as for CPI_type, at step S116, PlayItem which PlayItem_id points out in an EP_map type case acquires STC-sequence-id of AV stream to refer to.

[0422] In a control section 23, in an EP_map type case, at step S117, CPI_type inputs AV stream into a decoder based on above-mentioned STC-sequence-id and PTS of a mark. This STC-sequence-id and PTS of a marking point are used, and, specifically, it is step S54 of drawing 104 , The same processing as step S55 is performed. Or in a control section 23, in a TU_map type case, CPI_type inputs AV stream into a decoder based on ATS of a mark. This ATS is used and, specifically, it is step S54 of drawing 104 , The same processing as step S55 is performed.

[0423] A control section 23 makes a display start from the picture of PTS of a marking point at step S118, when CPI_type is an EP_map type. Or a control section 23 makes a display start from the picture after ATS of a marking point, when CPI_type is a TU_map type.

[0424] Thus, it carries out, as shown in drawing 106 , and a user chooses the start point of a favorite scene etc. from PlayList, and a recorder (control section 23 of the record regenerative apparatus 1) manages it to PlayListMark. Moreover, it carries out, as shown in drawing 107 , and a playback start point is chosen from the list of marking points that the user is stored in PlayListMark, and a player starts playback from the start point.

[0425] The content of the data currently recorded on the record medium, playback information, etc. can be managed appropriately, and it has them, and a user can check the content of the data currently recorded on the record medium appropriately at the time of playback, or it can make it possible to reproduce desired data simple by being based on such syntax, DS, and a regulation.

[0426] When the location of I picture can be analyzed and the location of I

picture cannot be analyzed using EP_map, it becomes possible to record, reproduce and manage AV stream of a different format to the same record medium with a common application program (software) by using TU_map.

[0427] When the contents (location of I picture) are analyzed and AV stream is recorded on a record medium (when carrying out cog NIZANTO record), TU_map is used, and EP_map is used when recording on a record medium as it is, without analyzing the contents (location of I picture) (when carrying out non cog NIZANTO record). With a common application program It can record on the same record medium, and can reproduce, and AV data can be managed.

[0428] When recording on a record medium as it is, without using and descrambling TU_map when descrambling AV data which followed, for example, were scrambled and recording them on a record medium (analyzing) (** which is not analyzed), EP_map is used, with a common application program, it can record on the same record medium, and can reproduce, and AV data can be managed.

[0429] Furthermore, TU_map can be used, when the location of I picture can be analyzed and the location of I picture cannot analyze EP_map type and TU_map type as CPI_type using EP_map, since it enabled it to describe in PlayLyst(). It enables this to unify and manage AV stream data which analyze and record the location of I picture, and AV stream data recorded without analyzing by the common program only by setting up a flag.

[0430] Moreover, by edit etc., since it dissociates independently and a PlayList file and a Clip Information file are recorded, when the content of a certain PlayList and Clip is changed, it is not necessary to change other files which are unrelated to the file. Therefore, time amount which can change the content of the file easily and the modification and record take can be made small.

[0431] Furthermore, only Info.dvr is read first, the content of record of a disk is shown to a user interface, and if only the PlayList file in which the user did playback directions, and the Clip Information file relevant to it are read from a disk, a user's latency time can be made small.

[0432] If all PlayList files and Clip Information files are collectively recorded on

one file, the file size will become very large. Therefore, the time amount which it takes in order to change the content of the file and to record it becomes very large compared with the case where dissociate independently and each file is recorded. This invention solves this problem.

[0433] Although a series of processings mentioned above can also be performed by hardware, they can also be performed with software. When performing a series of processings with software, the program which constitutes the software is installed in a general-purpose personal computer etc. from a record medium possible [performing various kinds of functions] by installing the computer built into the hardware of dedication, or various kinds of programs.

[0434] As shown in drawing 108 , this record medium is distributed apart from a computer in order to provide a user with a program. The magnetic disk 221 (a floppy disk is included) with which the program is recorded, an optical disk 222 (CD-ROM (Compact Disk-Read Only Memory) --) DVD (Digital Versatile Disk) is included. It is not only constituted by the package media which consist of a magneto-optic disk 223 (MD (Mini-Disk) is included) or semiconductor memory 224, but It consists of hard disks with which ROM202 with which a user is provided in the condition of having been beforehand included in the computer, and the program is remembered to be, and the storage section 208 are contained.

[0435] In addition, in this description, even if the processing serially performed according to the sequence that the step which describes the program offered by the medium was indicated is not of course necessarily processed serially, it is a juxtaposition thing also including the processing performed according to an individual.

[0436] Moreover, in this description, a system expresses the whole equipment constituted by two or more equipments.

[0437]

[Effect of the Invention] According to the record medium, like the above, in the 1st information processor of this invention and an approach, the program of a record medium, a program, and a list The 1st table which describes the response relation between a presentation time stump and the address in said

AV stream data of the access unit corresponding to it, Or one side of the 2nd table which describes the response relation between the arrival time stump based on transport Paquette's arrival time and the address in said AV stream data of transport Paquette corresponding to it was recorded according to the record approach.

[0438] According to the program, in the 2nd information processor of this invention and an approach, the program of a record medium, and a list The 1st table which describes the response relation between a presentation time stump and the address in said AV stream data of the access unit corresponding to it, Or the arrival time stump based on transport Paquette's arrival time, One side of the 2nd table which describes response relation with the address in said AV stream data of transport Paquette corresponding to it reproduces it from the record medium currently recorded according to the record approach, and controlled the output.

[0439] Moreover, the 1st information which shows the main playback pass according to the 2nd record medium, and the playback assignment information constituted using the 2nd information which shows the playback pass of ** reproduced synchronizing with said main playback pass were recorded on the 3rd information processor of this invention and the approach, the program of a record medium, the program, and the list.

[0440] The 1st information which shows the main playback pass according to the program, and the playback assignment information constituted using the 2nd information which shows the playback pass of ** reproduced synchronizing with said main playback pass are reproduced from a record medium in the 4th information processor of this invention and an approach, the program of a record medium, and a list, and the output was controlled based on it.

[0441] Therefore, in the case of which, AV stream in which high-speed playback is possible, and impossible AV stream are manageable in common. Moreover, after recording becomes possible.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1] It is drawing showing the configuration of the gestalt of 1 operation of the record regenerative apparatus which applied this invention.

[Drawing 2] It is drawing explaining a format of the data recorded on a record medium with the record regenerative apparatus 1.

[Drawing 3] It is drawing explaining Real PlayList and Virtual PlayList.

[Drawing 4] It is drawing explaining creation of Real PlayList.

[Drawing 5] It is drawing explaining deletion of Real PlayList.

[Drawing 6] It is drawing explaining assemble editing.

[Drawing 7] It is drawing explaining the case where subpass is prepared in Virtual PlayList.

[Drawing 8] It is drawing explaining modification of the playback sequence of PlayList.

[Drawing 9] It is drawing explaining the mark on PlayList, and the mark on Clip.

[Drawing 10] It is drawing explaining a menu thumbnail.

[Drawing 11] It is drawing explaining the mark added to PlayList.

[Drawing 12] It is drawing explaining the mark added to a clip.

[Drawing 13] It is drawing explaining the relation of PlayList, Clip, and a thumbnail file.

[Drawing 14] It is drawing explaining directory structure.

[Drawing 15] It is drawing showing the syntax of info.dvr.

[Drawing 16] It is drawing showing the syntax of DVR volume.

[Drawing 17] It is drawing showing the syntax of Resumevolume.

[Drawing 18] It is drawing showing the syntax of UIAppInfovolume.

[Drawing 19] It is drawing showing the table of Character set value.

[Drawing 20] It is drawing showing the syntax of TableOfPlayList.

[Drawing 21] It is drawing showing other syntax of TableOfPlayList.

[Drawing 22] It is drawing showing the syntax of MakersPrivateData.

[Drawing 23] It is drawing showing the syntax of xxxxx.rpls and yyyyy.vpls.

[Drawing 24] It is drawing explaining PlayList.

[Drawing 25] It is drawing showing the syntax of PlayList.

[Drawing 26] It is drawing showing the table of PlayList_type.

[Drawing 27] It is drawing showing the syntax of UIAppinfoPlayList.

[Drawing 28] It is drawing explaining the flag in the syntax of UIAppinfoPlayList shown in drawing 27 .

[Drawing 29] It is drawing explaining PlayItem.

[Drawing 30] It is drawing explaining PlayItem.

[Drawing 31] It is drawing explaining PlayItem.

[Drawing 32] It is drawing showing the syntax of PlayItem.

[Drawing 33] It is drawing explaining IN_time.

[Drawing 34] It is drawing explaining OUT_time.

[Drawing 35] It is drawing showing the table of Connection_Condition.

[Drawing 36] It is drawing explaining Connection_Condition.

[Drawing 37] It is drawing explaining BridgeSequenceInfo.

[Drawing 38] It is drawing showing the syntax of BridgeSequenceInfo.

[Drawing 39] It is drawing explaining SubPlayItem.

[Drawing 40] It is drawing showing the syntax of SubPlayItem.

[Drawing 41] It is drawing showing the table of SubPath_type.

[Drawing 42] It is drawing showing the syntax of PlayListMark.

[Drawing 43] It is drawing showing the table of Mark_type.

[Drawing 44] It is drawing explaining Mark_time_stamp.

[Drawing 45] It is drawing showing the syntax of zzzzz.clip.

[Drawing 46] It is drawing showing the syntax of ClipInfo.

[Drawing 47] It is drawing showing the table of Clip_stream_type.

[Drawing 48] It is drawing explaining offset_SPN.

[Drawing 49] It is drawing explaining offset_SPN.

[Drawing 50] It is drawing explaining the STC section.

[Drawing 51] It is drawing explaining STC_Info.

[Drawing 52] It is drawing showing the syntax of STC_Info.

[Drawing 53] It is drawing explaining ProgramInfo.

[Drawing 54] It is drawing showing the syntax of ProgramInfo.

[Drawing 55] It is drawing showing the syntax of VideoCondingInfo.

[Drawing 56] It is drawing showing the table of Video_format.

[Drawing 57] It is drawing showing the table of frame_rate.

[Drawing 58] It is drawing showing the table of display_aspect_ratio.

[Drawing 59] It is drawing showing the syntax of AudioCondingInfo.

[Drawing 60] It is drawing showing the table of audio_coding.

[Drawing 61] It is drawing showing the table of audio_component_type.

[Drawing 62] It is drawing showing the table of sampling_frequency.

[Drawing 63] It is drawing explaining CPI.

[Drawing 64] It is drawing explaining CPI.

[Drawing 65] It is drawing showing the syntax of CPI.

[Drawing 66] It is drawing showing the table of CPI_type.

[Drawing 67] It is drawing explaining video EP_map.

[Drawing 68] It is drawing explaining EP_map.

[Drawing 69] It is drawing explaining EP_map.

[Drawing 70] It is drawing showing the syntax of EP_map.

[Drawing 71] It is drawing showing the table of EP_type values.

[Drawing 72] It is drawing showing the syntax of EP_map_for_one_stream_PID.

[Drawing 73] It is drawing explaining TU_map.

[Drawing 74] It is drawing showing the syntax of TU_map.

[Drawing 75] It is drawing showing the syntax of ClipMark.

[Drawing 76] It is drawing showing the table of mark_type.

[Drawing 77] It is drawing showing the table of mark_type_stamp.

[Drawing 78] It is drawing showing the syntax of menu.thmb and mark.thmb.

[Drawing 79] It is drawing showing the syntax of Thumbnail.

[Drawing 80] It is drawing showing the table of thumbnail_picture_format.

[Drawing 81] It is drawing explaining tn_block.

[Drawing 82] It is drawing explaining the structure of the transport stream of DVR MPEG 2.

[Drawing 83] It is drawing showing the recorder model of the transport stream of DVR MPEG 2.

[Drawing 84] It is drawing showing the player model of the transport stream of DVR MPEG 2.

[Drawing 85] It is drawing showing the syntax of source packet.

[Drawing 86] It is drawing showing the syntax of TP_extra_header.
[Drawing 87] It is drawing showing the table of copy permission indicator.
[Drawing 88] It is drawing explaining seamless connection.
[Drawing 89] It is drawing explaining seamless connection.
[Drawing 90] It is drawing explaining seamless connection.
[Drawing 91] It is drawing explaining seamless connection.
[Drawing 92] It is drawing explaining seamless connection.
[Drawing 93] It is drawing explaining the overlap of an audio.
[Drawing 94] It is drawing explaining the seamless connection using BridgeSequence.
[Drawing 95] It is drawing explaining the seamless connection which does not use BridgeSequence.
[Drawing 96] It is drawing showing a DVR STD model.
[Drawing 97] It is the timing chart of decode and a display.
[Drawing 98] It is drawing showing the syntax of a PlayList file.
[Drawing 99] It is drawing showing the syntax of UIAppInfoPlayList in the PlayList file of drawing 98 .
[Drawing 100] It is drawing showing the syntax of PlayList() in the PlayList file of drawing 98 .
[Drawing 101] It is drawing showing the syntax of SubPlayItem.
[Drawing 102] It is a flow chart explaining the creation approach of Real PlayList.
[Drawing 103] It is a flow chart explaining the creation approach of Virtual PlayList.
[Drawing 104] It is a flow chart explaining the playback approach of PlayList.
[Drawing 105] It is a flow chart explaining the playback approach of the Sub pass of PlayList.
[Drawing 106] It is a flow chart explaining the creation approach of PlayListMark.
[Drawing 107] It is a flow chart explaining the search playback approach which used PlayListMark.
[Drawing 108] It is drawing explaining a medium.
[Description of Notations]

1 Record Regenerative Apparatus 11 thru/or 13 Terminal, 14 Analysis section 15 AV encoder, 16 Multiplexer 17 A switch, 18 Multiplexing stream analysis section 19 sow spa KETTAIZA 20 The ECC coding section, 21 Modulation section 22 The write-in section, 23 Control section 24 A user interface, 26 Demultiplexer 27 AV decoder 28 Read-out section 29 recovery sections 30 ECC decode section 31 Sow spa KETTAIZA 32 33 Terminal